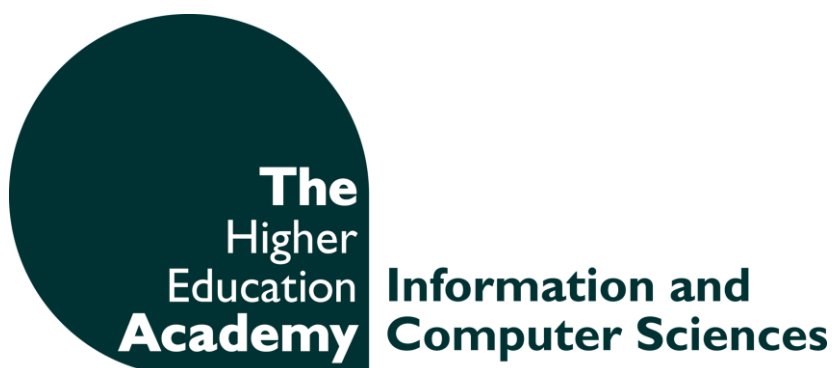




8TH INTERNATIONAL WORKSHOP ON TEACHING, LEARNING AND ASSESSMENT OF DATABASES (TLAD)



FOREWORD

This is the eighth in the series of highly successful international workshops on the Teaching, Learning and Assessment of Databases (TLAD 2010), which once again is held as a workshop of BNCOD 2010 - the 27th International Information Systems Conference. TLAD 2010 is held on the 28th June at the beautiful Dudhope Castle at the University of Abertay Dundee, just before BNCOD, and hopes to be just as successful as its predecessors.

The teaching of databases is central to all Computing Science, Software Engineering, Information Systems and Information Technology courses, and this year, the workshop aims to continue the tradition of bringing together both database teachers and researchers, in order to share good learning, teaching and assessment practice and experience, and further the growing community amongst database academics. As well as attracting academics from the UK community, the workshop has also been successful in attracting academics from the wider international community, through serving on the programme committee, and attending and presenting papers.

This year, the workshop includes an invited talk given by Richard Cooper (of the University of Glasgow) who will present a discussion and some results from the Database Disciplinary Commons which was held in the UK over the academic year. Due to the healthy number of high quality submissions this year, the workshop will also present seven peer reviewed papers, and six refereed poster papers. Of the seven presented papers, three will be presented as full papers and four as short papers. These papers and posters cover a number of themes, including: approaches to teaching databases, e.g. group centered and problem based learning; use of novel case studies, e.g. forensics and XML data; techniques and approaches for improving teaching and student learning processes; assessment techniques, e.g. peer review; methods for improving students abilities to develop database queries and develop E-R diagrams; and e-learning platforms for supporting teaching and learning.

We would like to thank members of the programme and steering committees for their reviews and their continuing support of this workshop. Many members have been involved in the workshops since the first TLAD, thus showing the strength of the database teaching community both within the UK and overseas. We would also like to thank the BNCOD steering and programme committees, who have ensured that once again TLAD has its place in BNCOD this year. Finally, we express our appreciation to the Higher Education Academy, and especially Karen Fraser, for her continuing support and efforts in assisting with the organization of the workshop past and present, and for the continued support by the HEA which is crucial to the success and future growth of TLAD.

Petra Leimich and David Nelson
Workshop Chairs

STEERING COMMITTEE

Petra Leimich (University of Abertay Dundee) Workshop co-chair
David Nelson (University of Sunderland) Workshop co-Chair
Richard Cooper (University of Glasgow)
Anne James (Coventry University)
Alastair Monger (Southampton Solent University)

PROGRAMME COMMITTEE

Les Ball, University of Abertay Dundee
Fang Fang Cai, London Metropolitan University
Jackie Campbell, Leeds Metropolitan University
Richard Cooper, University of Glasgow

Karen Davis, University of Cincinnati
Barry Eaglestone, University of Sheffield
Mary Garvey, University of Wolverhampton
Caron Green, University of Sunderland
Anne James, Coventry University
Petra Leimich, University of Abertay Dundee
Nigel Martin, Birkbeck University of London
Pirjo Moen, University of Helsinki
Alastair Monger, Southampton Solent University
David Nelson, University of Sunderland
Mick Ridley, University of Bradford
John Wilson, University of Strathclyde

HIGHER EDUCATION ACADEMY

Karen Fraser (University of Ulster)

CONTENTS

INVITED PAPER:

A Disciplinary Commons for Database Teaching

Les Ball (*University of Abertay*), **Sheila Baron** (*Southampton Solent University*), **Charles Boisvert** (*Norwich City College*), **Richard Cooper** (*University of Glasgow*), **Tugrul Essendal** (*De Montfort University*), **Sally Fincher** (*University of Kent*), **Tony Jenkins** (*University of Leeds*), **Petra Leimich** (*University of Abertay*), **Al Monger** (*Southampton Solent University*), **David Nelson** (*University of Sunderland*), **Thomas Neligwa** (*Keele University*), **James Paterson** (*Glasgow Caledonian University*), **Clare Stanier** (*Staffordshire University*), **Tony Valsamidis** (*University of Greenwich*), **John Wilson** (*Strathclyde University*)

Page 5

FULL PAPERS:

LeReSpo: Using Game Software in Database Education on all Levels

Carsten Kleiner (*University of Applied Sciences and Arts Hannover*)

Page 9

Uses of Peer Assessment in Database Teaching and Learning

James Paterson (*Glasgow Caledonian University*), **John Wilson** (*University of Strathclyde*), **Petra Leimich** (*University of Abertay Dundee*)

Page 18

SQL Patterns - A New Approach for Teaching SQL

Huda Al Shuaily, Karen Renaud (*University of Glasgow*)

Page 28

SHORT PAPERS:

A Relational Algebra Tutor

David Nelson, Jonathan Scott (*University of Sunderland*)

Page 40

Making the Most of an e-Learning Platform to Promote Collaborative Learning: A Case Study

Liz Sokolowski (*Thames Valley University*)

Page 42

An Entity-Relationship Diagram (ERD) Marking Scheme

Paul Henderson (*Sheffield Hallam University*)

Page 48

Teaching Undergraduate Students Data Mining: Ideas, Experience and Challenges

Hongbo Du (*University of Buckingham*)

Page 49

POSTER PAPERS:

Introducing a Forensic Flavour to Teaching Databases at L5

Jackie Campbell, Sanela Lazarevski (*Leeds Metropolitan University*)

Page 54

Exploring the XML-Relational Interface using Problem Based Learning

Peter Lake (*Sheffield Hallam University*)

Page 58

Handling XML Data using Oracle 11g

Mary Garvey (*University of Wolverhampton*)

Page 62

Teaching and Learning Databases using a Group Centred Interactive Problem Solving Approach

Raman Adaikkalavan (*Indiana University South Bend*)

Page 67

Errors in Entity-Relationship-Diagrams: Classification and Statistical Data

Martin Herzberg (*Martin-Luther-University, Halle-Wittenberg*)

Page 72

A Completing Application for a Database Coursework Assignment

Richard Cooper (*University of Glasgow*)

Page 77

A DISCIPLINARY COMMONS FOR DATABASE TEACHING

LES BALL (UNIVERSITY OF ABERTAY), SHEILA BARON (SOUTHAMPTON SOLENT UNIVERSITY), CHARLES BOISVERT (NORWICH CITY COLLEGE), RICHARD COOPER (UNIVERSITY OF GLASGOW), TUGRUL ESSENDAL (DE MONTFORT UNIVERSITY), SALLY FINCHER (UNIVERSITY OF KENT), TONY JENKINS (UNIVERSITY OF LEEDS), PETRA LEIMICH (UNIVERSITY OF ABERTAY), AL MONGER (SOUTHAMPTON SOLENT UNIVERSITY), DAVID NELSON (UNIVERSITY OF SUNDERLAND), THOMAS NELIGWA (KEELE UNIVERSITY), JAMES PATERSON (GLASGOW CALEDONIAN UNIVERSITY), CLARE STANIER (STAFFORDSHIRE UNIVERSITY), TONY VALSAMIDIS (UNIVERSITY OF GREENWICH), JOHN WILSON (STRATHCLYDE UNIVERSITY)

ABSTRACT

This paper discusses the experience of taking part in a disciplinary commons devoted to the teaching of database systems. It will discuss the structure of a disciplinary commons and our experience of the database version.

Keywords

disciplinary commons, teaching practice.

1. INTRODUCTION

A Disciplinary Commons is an attempt by teaching professionals to come together and share teaching practice and experience. This is achieved through a series of monthly meetings in which all aspects of teaching one particular course is analysed from the context of the course thorough to evaluation. By picking one course, the participant can reflect on how the teaching is organised, what is taught and how effective it seems to be. It is clearly useful as a means of developing teaching skills, but can also be useful for documenting practice.

The Disciplinary Commons [1, 2] initiative is led by Josh Tenenberg in the USA and Sally Fincher in the UK and takes a group of participants by focussing at each meeting on one aspect of the course: the teacher; the context in which the course operates; the content of the course; the instructional design of the course; the way the students are assessed; how we evaluate the success or otherwise of the course; and how the course is delivered.

The structure involves a great deal of peer evaluation. This includes paired observation of teaching – for instance giving a lecture or a tutorial; comment on the aspects of the portfolio as it is developed; and general discussion of each issue. The outcome is a portfolio which includes course artefacts and commentary from the individual [3].

In the UK, successful disciplinary commons have been run for introductory programming [4] and HCI (both accessible from [1]) and this year we have run one for the teaching of database systems. The paper describes the structure and then the experience of the database commons.

2. THE STRUCTURE OF A DISCIPLINARY COMMONS

A disciplinary commons starts by attracting a group of individuals interested in exploring their teaching practice in a particular area. In the area of database, this was somewhat made easier by the pre-existence of such a group – the participants of previous TLAD workshops.

The work consists of attending nine monthly meetings as follows:

1. An introductory meeting in which participants get to know each other, describing how they got involved in teaching and their teaching ethos. Pairs for the observation of teaching practice are set up.
2. A meeting which discusses the context of the course being documented – i.e. how it fits into the overall programmes to which it contributes, the nature of the students and the departmental teaching ethos.
3. The third meeting discusses content. The material used, textbooks recommended and the aspects of the discipline which make up the syllabus are all under discussion at this meeting.
4. Instructional design is the topic of the fourth meeting. This involves the balance of lectures and labs, which material is taught by what method as well as which tasks the students are set.

5. There then follows a discussion of assessments – exams or coursework and the mapping of intended learning outcomes to assessment methods.
6. The sixth meeting has course evaluation as its focus. How do we determine whether the course has been successful? Student feedback and student results are the main mechanisms of course, but the meeting also looks at how the institution values and makes use of any evaluation.
7. Delivery mechanisms are the basis of the seventh meeting, but by now focus should be shifting to the portfolio being developed. At this stage, mutual assessment of the material being developed for the portfolio should be beginning to take place. A series of pairings to give feedback on the material are started about now.
8. At this meeting, a draft portfolio should be available and this is discussed with an observing partner.
9. At the final meeting, participants get to display their portfolios and get final feedback.

3. THE EXPERIENCE OF THE DATABASE COMMONS

The commons was advertised firstly through TLAD and BNCOD in July 2009 and also through the HEA mailing lists. About twenty people expressed interest, but after checks with commitments, the number stabilised at fourteen. It was decided early on to make this a peripatetic enterprise. Whereas both the programming and HCI commons had run in London, we decided to share the hosting (and the travelling), particularly important as six of the fourteen were from Scotland or the North-East and only three were in the South East. In the event, a severe winter hampered travel in the middle of the year, but even so no-one failed to make more than two meetings and so the three-strikes and you're out rule was not needed.

We started and ended with a meeting in Glasgow. The introductory meeting had each of us introduce ourselves and this was very illuminating because none of us had taken the school – university – research – lecturer role. Rather, we had mostly had previous jobs outside the University sector and then moved into non-lecturing jobs before evolving into lecturers, each in our own way. This set the tone for a high degree of communality for the year, despite the wide variety of institutions involved from Russell group universities to an FE college.

The next two meetings in Greenwich and Abertay concentrated on the context and content of the course. Here it was clearly that there were wide differences both in the classes we were teaching and the expected content. There were first year courses, later undergraduate courses and courses for Masters students – often introductory courses for "generalist" programmes. We drew context maps to show how the course fitted within the overall programme(s) that the students were taking. Many of us were teaching, either in the same course or more likely a different one, some form of internet programming. This in turn influenced the context of the course and the degree to which basic theory could be covered.

There were consequently differences in content from very basic database design / SQL querying, through to more thorough treatment of database principles. There was much debate at Abertay about how to teach normalisation and relational algebra or whether it is possible to do so. It had been our intention to pull together our opinions on the usefulness of various course materials, such as textbooks, but this was never achieved due to time pressure.

One of the most significant findings of the disciplinary commons was how the database curriculum has been gradually and systematically eroded at all levels in order to accommodate various external factors, such as the lack of teaching resources and the pressure to keep up-to-date with new technological developments.

Database modules at many universities have been squeezed in with other topics such as Web programming, human computer interaction, systems design, and even computer graphics. As a result, teaching has generally been limited to basic database design and SQL programming skills with little or no theoretical foundation. The absence of theoretical concepts and mathematical formalisms is a cause for concern because their absence in formative years will eventually have an adverse effect not only on the development of sound database systems but also on the database research in the future. A similar viewpoint was expressed in [5] which warns about the negative consequences of a recipe-oriented approach to module design and learning without a sound theoretical framework and argues that the module placement in the programmes is key for creating the space that is needed for covering core database concepts.

We then turned to instructional design at Leeds in January. Whereas most of us used the traditional mix of lectures and labs, there was some difference in the degree of support and the balance, with one institution now using no lectures at all.

Assessment by coursework and exam, also discussed in Leeds at the next meeting, was carried out in the usual way, but some courses were assessed purely by coursework and there was considerable debate between those who felt that the basic concepts of database management could only be genuinely assessed

away from the support environment of the DBMS and those who felt that the practical use of a DBMS is what was the most important thing to teach and so coursework was sufficient to assess this.

The ways we evaluated were discussed in Southampton and what we had achieved again used much the same mechanisms – student feedback through questionnaire, staff student meetings etc. - to tell us what the students think. Assessment results tell us how well they actually absorbed the material. Personal reflection and preparation of the same material for next year leads to ways of doing things differently, hopefully better.

Another aspect of this was the expected formal evaluation required by the institution, discussed back in Greenwich. Most had some form of reporting mechanism, but this could range from a short report given to the programme director (what had changed, did it go well, do the marks need moderation, etc.) to central university driven reports whose ultimate use could only be guessed at. One pairing threw up the contrast between one institution preparing a narrative about their course independently of any other, from another in which the report is in the context of the programme as a whole.

The portfolios were developed much more slowly than is really desirable and the penultimate meeting in Sunderland mostly discussed plans for the final portfolio. The impact of other work, notably exam preparation and coursework and exam marking, made the time when this work should be achieved disappear all too easily. A few people did good early work, but many had still not produced much of a portfolio for the eighth meeting, although AI had achieved a virtually complete portfolio of high quality by that time as had some others, while most had much less to show. This has been written between the eighth and ninth meeting and it is expected that everyone will have a complete portfolio by the final meeting.

4. SUMMARY

We have all found the exercise very successful and valuable, each in our own way. The enterprise is certainly a stressful addition to an already busy working life and few of us managed a timely development of our portfolios. This, in turn, reduced the amount (and therefore the value) of peer evaluation, although there was enough of this that we all felt that this was the most important benefit of the exercise.

From our discussions, we did share many techniques and tricks used by others which we found to be valuable in our own teaching. Among these from significant technological tools, such as the use of Peerwise [6] to test students by seeing if they could formulate interesting quiz questions, to Richard's adopting Charles demonstration of the need to master four different languages to create a dynamic web page by juggling an increasing number of balls, although he feels he did this in a more authentic manner (a) because he can't juggle, and (b) because he used four completely different objects.

We also found, as we probably expected, the challenges, and the ways we deal with them, to be very similar whatever the institution and cohort being taught. Furthermore, the ability to step outside of our usual mechanisms for self-evaluation was of immense benefit. Whereas, formal university evaluation mechanisms specify what is to be evaluated and how, the commons allowed us to interpret our practice as we found helpful and relevant and enabled us to explore aspects that we might not otherwise have been considered.

Our web site can be found from <http://www.cs.kent.ac.uk/people/staff/saf/dbdc/>.

References

- [1] Pat Hutchins (ed.), *Making Teaching Community Property: A Menu for Peer Collaboration and Review*. American Association for Higher Education, Washington, D.C., USA, 1996.
- [2] The main disciplinary commons web site is <http://www.disciplinarycommons.org/>
- [3] Pat Hutchings (ed.), *The Course Portfolio: How Faculty Can Examine Their Teaching to Advance Practice and Improve Student Learning*, American Association for Higher Education, 1998.
- [4] Sally Fincher, David Barnes, Pete Bibby, James Bown, Vicky Bush, Phil Campbell, Quintin Cutts, Stephan Jamieson, Tony Jenkins, and Michael Jones. *Some good ideas from the disciplinary commons*. In *Proceedings of 7th Annual Conference of the ICS HE Academy*, pages 153-158, August 2006.
- [5] M.J.Ridley (2003), *Database Systems Or Database Theory- Or "Why Don't You Teach Oracle"*, TLAD, Coventry (2003).
- [6] Paul Denny , Andrew Luxton-Reilly , John Hamer, The PeerWise system of student contributed assessment questions, *Proceedings of the tenth conference on Australasian computing education*, p.69-74, January 01-01, 2008, Wollongong, NSW, Australia.

LEReSpo: USING GAME SOFTWARE IN DATABASE EDUCATION ON ALL LEVELS

Carsten Kleiner

University of Applied Sciences & Arts Hannover

Ricklinger Stadtweg 120

30459 Hannover

Germany

ckleiner@acm.org

<http://www.fakultaet4.fh-hannover.de/fakultaet-iv/index.html>

ABSTRACT

In this paper the software LeReSpo is introduced which provides games for improving the learning process in undergraduate database system courses. It features two different types of games that may be used to enforce knowledge about database topics. These games can be used in many different scenarios and thus provide a very flexible tool to check database knowledge for students in class and at home. Questions to be answered in these games may contain multimedia elements such as video clips to enhance the fun factor for students. The system also provides a management frontend for instructors to manage games and questions to be used in those games. Finally, as the system internally uses an object-oriented database system to manage games and questions, it is also suited to teach advanced database concepts in graduate classes. This is on one hand done by providing a lively example for the usage of OODBS. On the other hand it offers exercises and projects for enhancing the database component of the system. Future improvements of the current system are explained along with initial results from using it. The system may also be used for similar purposes in different domains.

Keywords

Object-oriented databases, game software, database education, master class, information systems, jeopardy.

5. INTRODUCTION AND MOTIVATION

Student interest in database classes has significantly been declining in recent years. I have observed this phenomenon in our institution by poor student attendance in practice sessions, low student motivation when working on database exercises as well as diminishing interest in all topics related to database systems. Moreover poor student performance in database system exams (probably induced by the problems explained before) and also reduced interest in classical database topics as thesis subjects shows that database classes seem to be anything but en vogue at the moment.

Fortunately this has not only been observed at my school, but has been reported in many other publications from colleges and universities at all levels and from different countries (e.g. [6], [7], [22]). Since most educators (and practitioners as well) still consider database systems to be an integral and very important part of computer science education, it is necessary to tackle such problems in order to be able to create well employable graduates.

Apart from raising awareness for the importance of database topics for their future career (which unfortunately only rarely increases interest in database classes in my experience) I believe there are two important points that we can work on in order to increase student motivation:

- **Showing the practical relevance of database system topics in real-world examples of large software systems:** Students typically consider themselves developing large software systems in the future. If we can manage to make them believe that database systems are an important part of many large-scale software systems by presenting them appropriate real-world examples, we might be able to raise the level of interest in database classes significantly.
- **Increasing the fun level in database system classes:** Database system classes are typically known

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

as being rather analytical, sober and unemotional. This is partially due to the solid theoretical foundation of relational database systems which are a typical part of classical database system classes. In addition, the analytic skills required succeeding in modeling and mastering database systems are typically considered difficult. Therefore we should try to embed the theoretical and analytical topics in a more up-to-date and enjoyable setting. This might lead to an increased interest in database topics, even if they are difficult.

In this paper I will present a software system that has been developed and used at my school to address both aspects mentioned previously. The main idea of the system LeReSpo (a german acronym corresponding to the application domain for which it had originally been designed) is to provide a software system that can be used to play a Jeopardy-like quiz show. The system features a grid of questions on different topics and difficulty levels to be used in a two team competition. The questions are hidden and revealed only upon choice by either team. The team with more points earned based on correctly answered questions wins. In addition, a single team mode (called question trainer) is also provided as a second game. The questions themselves can be of different types and optionally with multimedia elements to further enhance the fun factor. More details about the system are explained in section 3. The system can be used in different situations within an undergraduate database class, e.g. for exam preparation or just to add a fun element in class.

Moreover the system has been partially developed, migrated and extended in a graduate information system class in order to introduce object oriented database systems. The direct practical application and the ease of use of the underlying OODBS greatly improved interest in OODBS in the first place and in database systems in general as well. In future years the graduate students will already know the game from their own undergraduate classes and are likely to be keen on understanding the underlying software and further improve it based on their own experiences. The system is only in place for about 1 ½ years; therefore no significant results can be reported by now but some promising comments from students have already been collected.

The paper is organized as follows: after a review of related work in section 2 I will describe some more details on the concept and implementation of the LeReSpo software in section 3. Thereafter I will illustrate its use in undergraduate classes where the game aspect is more important. In section 5 I will briefly elaborate on the integration of this system into a graduate-level information systems class. Finally, I will conclude with some ideas for future improvements which might be topic of the graduate-level class next winter.

6. RELATED WORK

As already explained in the introduction nowadays it is generally accepted that database system classes receive less attention by students. This holds for computer science students as well as for non-majors. Also students tend to dislike these classes and feel forced to attend them. This has been reported in literature (e.g. [22] for Germany, [6] mainly for the USA, [1] globally) as well as in many informal discussions with different instructors.

In order to revert this process several approaches to enhance student motivation for CS in general, as well as for database system classes in particular, have been employed. In [7] several different approaches from the instructor community have been assembled. The paper [11] tried to enhance motivation by using a database practicum, whereas [10] already employed games in order to let students have more fun. The idea is very similar to mine except that they used different tools. The overview of existing tools to support database education discussed in [10] contains e.g. esql ([8]), WinRDBI ([5]), SQL-tutor ([13]), Kermit ([23]), Normit ([12]). Even though this reference is already seven years old the list of tools is still almost complete. Whereas all those tools focus on learning certain technological or conceptual aspects of database systems in detail, the software LeReSpo as it is does not directly improve upon the learning process. It rather provides a game tool for self- or group assessment where the outcome may improve the motivation to use one of the other tools in order to score better in the next instance of the game.

There have also been approaches to employ sophisticated tools to teach database systems. These tools are meant to increase the interest of students as well as to improve on the learning process itself. Examples of these can be found in [3], [9] and [20]; the focus of most of the tools is on the SQL language rather than more general database system concepts ([20] is an exception).

Many instructors see the reason for the disinterest in database systems in the analytical and theoretical background required to learn the concepts. This does not seem to fit the life style of an ever increasing online generation. To remedy this effect, many papers suggest using multimedia elements in teaching in order to pick up students of this generation at the right spot. Examples for these approaches are described in [17] and [25]. Papers [15] and [16] also describe using multimedia elements but the main focus there is on improving upon the learning process in general.

There has also been some work on including object-oriented database systems into the database curriculum (e.g. [4] and [18]) which in most cases is of a more conceptual nature. This is also true for most parts of [24]

which also contains a chapter on OODBS. The conceptual idea on including OODBS into a graduate class on database systems is not subject of this work. The game application LeReSpo rather yields an interesting and motivating example of using an OODBS in practice which may be used in the exercise section of such a class.

Parallel with the previously described development we observe an increasing interest and demand in e-Learning. This is specifically true for computer science students. Thus tools supporting e-Learning in database systems can already be found in the literature, e.g. [19], [21]. The tools described in [2] and [14] even go beyond database system classes. I will later describe how the software system described in this paper can also be used for e-Learning purposes in sections 3 and 6.

7. CONCEPTUAL DESIGN OF LERESPO

7.1 Conceptual Overview

The software system LeReSpo in its current form consists of three components as shown in Figure 1. These are the persistent store for all data required for the games, an administration frontend for the database as well as a player frontend.

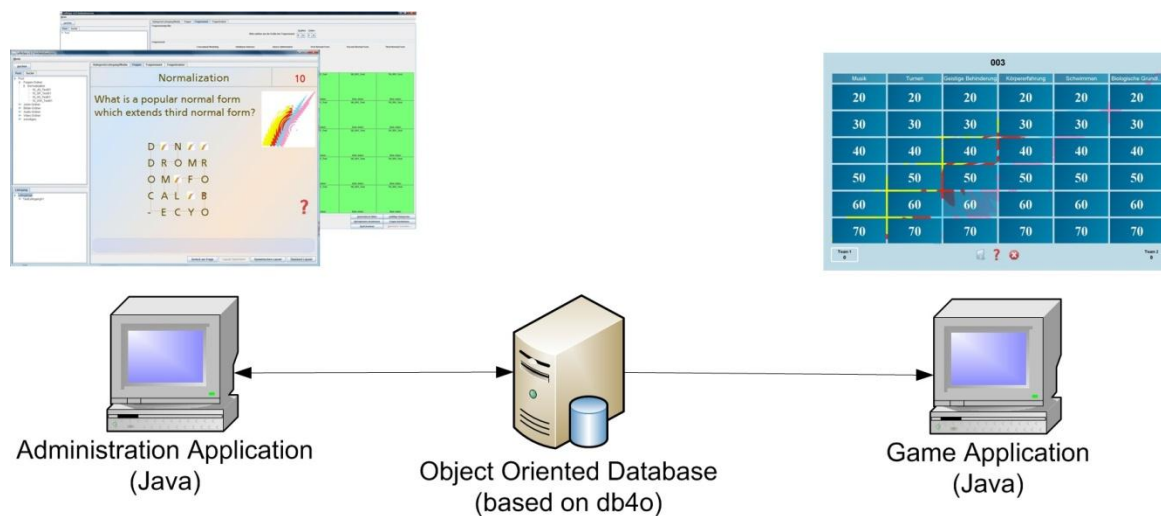


Figure 1: Conceptual Architecture of LeReSpo

Core of the system is an object-oriented database containing all data required for LeReSpo. The data model for this database is shown in Figure 2. The database which is roughly structured in courses, categories, games, media and questions is managed by a Java application for administrators, i.e. course instructors. This application is used to create, modify and delete all relevant data for the game software. The other major component is the player frontend which is used to actually play previously or dynamically created games consisting of questions in the database. Details on these applications are explained in section 3.2.

The main unit of organization within the database are courses (cf. Figure 2); courses may correspond to a full semesters course offering or to individual lectures as desired by the instructor. Courses contain the individually created games by the instructor which may be either two player Jeopardy-like walls or single player cardbox like games. Jeopardy walls consist of certain categories which are offered as column headers (cf. Figure 3).

Categories are used to organize questions thematically, where each question is assigned a unique category. Questions currently can be of any of five predefined types. Additionally they may contain media objects which may be either used for a sophisticated layout of the presentation of the question or be part of the question itself. E.g. a video animation of schema normalization may be presented and the question is which error has been made in the video. This also provides a handy interface on how to integrate external database courseware such as [15], [16]. Each media object may be related to different questions. Every question can have at most one video and audio assigned (as this is typically used for the task itself), but many different images may be assigned for the layout.

7.2 Using administration and player frontend

Initially after creation of a course and corresponding categories in which the questions may be divided, the instructor is able to upload the desired media objects which may be used for layout of questions as well as

integral parts of the questions, i.e. sound or video clips on which questions are related. Reuse of media elements is possible since they are managed separately from the questions and dynamically assigned by the instructor.

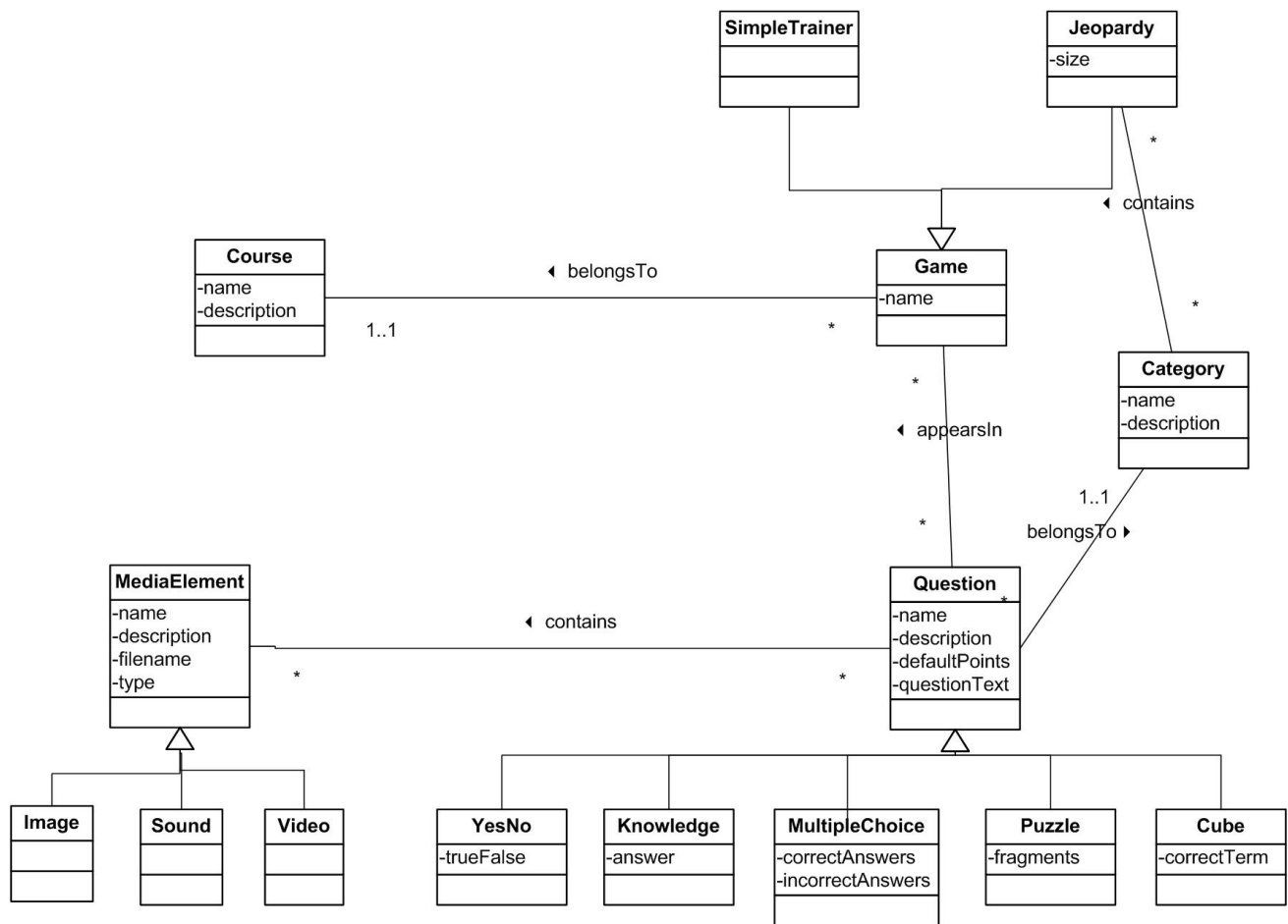


Figure 2: Data Model for Persistent Data (greatly simplified)

The instructor may then start creating the questions which may be chosen from certain predefined types. These types are true-false questions, knowledge questions (answer is a string), multiple choice (with potentially multiple correct answers) as well as cube (letters of a fixed term are arranged in a cube and the term has to be identified) or puzzle (a statement is separated into several different fragments of given size; the fragments have to be rearranged to reconstruct the statement). Example screenshots of these different types of questions are shown in Figure 3.

After creation of the questions the instructor may choose to create personal games using the categories and questions entered before. This is particularly appealing if certain aspects of the class need more in depth training than others or specific aspects seem to be misunderstood. Note that an automated as well as semi-automated generation of games based on a given set of categories is also supported. Automated generation means that both categories as well as questions in each category are chosen arbitrarily for a given grid size.

Semi-automated generation refers to a manual choice of categories and size of the game where the questions are then chosen automatically for the game.

After finishing the creation of games, in the current version, the complete database is transferred to the game playing application (also a Java application) in a deployment step. The game playing application may then be distributed and used by students or instructors. In the current version the database is replicated to each installation of the application for simplicity of administration and to provide offline capabilities. Future versions will probably directly access a single database server online, reducing the size of the game package and making the content more dynamic. This is also required to web-enable the games later.



Figure 3: Sample screenshots for true-false question (top left), knowledge question (top right), multiple choice question (middle left), cube (middle right) and puzzle (bottom) questions

Games can be either the two-team jeopardy-style games (cf. Figure 3) where a wall of categories is shown and questions of differing value are offered in each category to either team in turn. The other game option is for an individual player. It is offering a fixed set of questions in arbitrary order to the player one after the other. At the end of a game a graphical evaluation of the answers by category is shown, so that the students can identify their own potential weaknesses. Currently in both game applications answers to questions are automatically evaluated by the application and classified as correct or incorrect. This has the drawback that minor errors to answers with strings (e.g. typos) are classified as wrong; in the future we plan to include an optional manual classification of the answers at least for the jeopardy version. In that case the instructor using the game in class can be somewhat lenient on minor errors.

8. USING LERESPO IN UNDERGRADUATE CLASSES

The software system as explained in section 3 can be used in any undergraduate course on databases. The instructor uses the administration tool to assemble a catalogue of questions organized in categories




Jeopardy Demo Databases I					
Conceptual Modeling	Database Indexes	Query Optimization	First Normal Form	Second Normal Form	Third Normal Form
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40			40	40
50	50			50	50
Lions 10		 		Tigers 10	

Figure 4: Example of a two-team Jeopardy wall

corresponding to certain areas of knowledge in the course. He should also prepare a set of well-designed games which are capable of testing the most relevant issues for the particular course. These games may be used in different situations and for different purposes within a course:

- **Major interactive and fun element of a lecture:** A Jeopardy-like wall may be used for this purpose; the auditorium has to be subdivided into two groups which play as opponents. Either a group leader or every participant in turn decides on the next question to choose and is responsible for the answer. The wall should not be too large in order to keep time for the game limited and still be able to finish it. This application scenario only works in lectures of limited numbers of students (up to around 50).
- **Minor interactive element of a lecture:** The simple trainer game may be used for this purpose. A small number of dedicated questions is arranged in a game and is then presented to the class. Students may apply for answering the questions individually. It may be used e.g. towards the end of the lecture and every student getting an answer correct may leave the room. This enhances motivation on answering in general and also raises the concentration level. There has to be a sufficient number of questions for every student or the remaining group is released at once when all questions have been used. It can also be used within the lecture to achieve a change of media and more interactivity which is desirable from time to time. In this case rather small numbers of questions are typically used. This application scenario is almost independent of the size of the group.
- **Practice session:** The usage scenarios explained for lectures are also valid for practice sessions. Actually since the number of participants is typically much smaller, it is much easier to use the games in practice sessions. Teams can be real teams in this setting which may internally discuss solutions before they are finalized. The Jeopardy game format greatly benefits from smaller groups.
- **Support of e-Learning:** Since the games must be played offline in the current version where the database of questions is delivered with the game playing application, students can take the application home and use it wherever and whenever they want. This supports individual student preparation as well as e-Learning. Note that both forms of games are suitable for individual preparation, even though the Jeopardy wall seems biased towards teams. The fun factor is higher than for the question trainer which is why many students will even prefer the wall game when they play alone. Learning groups are also supported by this take home version of the games.
- **Exam preparation:** The games are helpful in any stage of exam preparation. They may be used in the final lecture or practice session to enforce particularly important topics by the instructor on one hand. On the other hand they are as helpful for individual student preparation of exams (may be some hints to the more relevant games by the instructor are appreciated) or for smaller student groups during self study phases.

One of the next steps in further development will be to design and implement a web-based game playing component which opens up new application scenarios such as “game of the week” and supports distance education.

Finally I would like to raise the attention to inherent weaknesses of the system. The format of questions and answers is only suitable for reinforcement of already existing knowledge; even in this context it should not be

used for certain tasks, e.g. data modeling, which cannot be reduced to the question and answer setting. This is somewhat diminished by the feature of using media files which could be used for more complex tasks. But it should be noted that there are certain types of knowledge and capabilities that cannot be improved upon by the system. Also the automated evaluation of answers might be critical: it is implemented based on a somewhat lenient string comparison. A more sophisticated algorithm would be very helpful here, particularly considering the possibility of student frustration. Imagine a student getting the answer correct but making a minor spelling error which leads to no credit being awarded.

9. USING LERESPO IN GRADUATE CLASSES

One of the very nice aspects of the software system described in this paper is that it provides a second application area: apart from being used as a playful learning tool in undergraduate education it may also be used in graduate classes. I offer a first semester course in our master's program on advanced database and information systems where e.g. non-relational databases are introduced. Since the game software is a perfect example for using an object-oriented database system, I included it into the class. Advantages of using an OODBS in this scenario are:

- Very small footprint of the embedded version of db4o which is used here
- Zero database administration because it is used in embedded mode
- Minimal persistence overhead: there is a single persistence class taking care of configuration of the database as well as storing and retrieving objects
- No impact on the entity classes in Java code: the implementation of the objects in the conceptual model (cf. Figure 2) could be done in a straight forward way without having the persistence in mind

Altogether using an OODBS as opposed to a relational database has just been simpler and easier to do in this context.

Therefore LeReSpo has been used as an example in the master's course to illustrate the benefits of OODBS in certain situations. By showing this application to the students they believed in the benefits of OODBS because the application could be easily understood on one hand and on the other hand was complex enough to be taken seriously. The class provided an in-depth analysis of the usage of the OODBS in the lecture. This analysis included the design of the persistence package on Java side which had to be stored in the OODB later. The students were instructed to carefully design their persistent classes because of the automated dependency persistence mechanisms in an OODBS; you have to be careful not to make your whole application (including GUI) persistent. Also queries have been introduced to retrieve objects of interest from the database to the application. In db4o queries have been expressed in either native code or in the query by example style which proved very intuitive to students.

The analysis in class was extended by practical exercises which had the goal to implement small extensions to the system or experiment with different configuration options of the database. In the future I plan to assign comparison projects with other OO database systems and/or with relational databases (plus persistence frameworks).

An embedded OODBS such as db4o typically operates schema-less. This seems to have the advantage of full flexibility for the application programmer (which is typically considered positive by students) in first place. On the other hand this leads to an increased complexity for the application programmer because there are no constraints on which to rely on. Thus the application has to be implemented much more fault tolerant than initially expected. There might even be new data types in the database which an old version of the application is not capable of processing. This should still result in a working application. Many students in undergraduate classes tend to dislike schemas and constraints because it takes them longer to achieve working applications. By showing the effects of a schema-less database such as in this case, we could significantly raise the level of acceptance for integrity constraints and schemas in our master class. This is an important learning process in my opinion in order to value the benefits of a schema. In the class these problems were shown by using different versions of the application on the same database. Also an extension to the original data model had to be implemented which required significant work on both applications.

10. CONCLUSION AND FUTURE WORK

In this paper I have described the motivation, concept and implementation of a learning support tool for database education. The tool provides students with two different types of games with which they can test their knowledge of database topics by answering certain predefined questions. Questions can be of different types with some focusing on a playful setting and others being more formal. The questions and games are stored in an OODBS which is managed by a second application for the instructors. I have also described usage scenarios for the games in undergraduate classes as well as for the system as a whole in graduate classes.

The goals of the system are two-fold: on one hand I hope to increase the fun factor in database classes to raise the level of interest of students. This has been steadily decreasing over the years and new approaches are required to remedy this process. Definitely games are a valid method for making classes more attractive (cf. [10], [25]). Therefore the system described is expected to be very helpful to this end. Initial observations from using it in two different database system courses support this claim qualitatively. Also the interest in OODBS in the master level class has been significantly improved by using LeReSpo as concrete example in class.

On the other hand I hope to increase the level of knowledge of database topics by students in general (to improve exam results and consequently the interest in database topics). This goal is supported by the games application as well. Firstly, students tend to learn better if different learning approaches are blended within a single class (such as games, lecture and traditional exercises). Moreover the games provide a good foundation for tool-assisted self-learning where students can study the topics whenever and wherever they want and still be supported by technology.

Altogether LeReSpo in its current state is already a very helpful tool supporting database education. But it should again be noted that naturally it is only a supporting tool. Its intension is not to provide a means of initially teaching new concepts. This has to be done in some other way. The tool can only verify knowledge which already exists. Furthermore, by using the question and answer format for the game, certain (very important) concepts cannot be enforced with the tool. In a database context this e.g. corresponds to conceptual modeling or detailed query optimization. But still a fair amount of the topics in a traditional database course may be trained by the Q&A style.

As of today the tool is very well usable but requires many extensions and changes to become even more efficient and helpful. The first and most important issue will be to get rid of the database duplication very every installation. The original development of the tool (which has been in a completely different domain, namely trainer instruction for disabilities sport) had the requirement of offline usage. This is somewhat outdated nowadays which is why we plan to migrate to a central standalone database server. This server is then used by both the instructor as well as the student applications. Since the OODBS used is available in an embedded as well as a server mode this migration should be rather simple. The next subsequent step would then be to web-enable the game playing application in order to get rid of the necessity for local installation on the student's computers at all. This is a more demanding change which will be implemented as a separate project.

Also it would be very beneficial to implement more types of questions (even with more gimmicks to enhance the fun factor even further) as well as more types of games. There are already initial ideas for both in place, but a concrete plan on how to implement them still has to be considered future work.

Regarding the educational perspective I would like to use the existing games and questions in more classes and perform a structured assessment. This could help in quantitatively proving the effectiveness of the tool. Also this could be used to get more input from the students for further improvements. Currently the tool has only been used in very few classes and therefore a quantitative student feedback has not been collected yet.

In order to base the experiences and ideas for improvements on a broader range of experience using the game tool in different classes in computer science is an option. It should be pointed out that the current tool is by no means restricted to being used in database classes (it had originally been developed for a completely different domain) regarding the gaming aspect for undergraduate classes. Therefore the number of classes where the tool could be used is big. Even considering the initial amount of work that is required to set up a reasonable pool of questions I expect to convince a couple of colleagues to experiment with the tool in other computer science classes as well. Coupled with a quantitative analysis and evaluation this could support the claim of usefulness. It might actually prove more useful in one class than in others.

Finally in terms of eLearning which is clearly improved by the tool it would be extremely interesting and promising to couple the game with existing approaches for learning tutors. Instead of offering fixed sets of questions to the students the system could adapt to the performance of the particular student or group and offer tailored questions which are particularly important for the given level of knowledge. The system could as result of a completed game automatically suggest other games that might be of particular interest to the student in order to improve his specific skills (or missing knowledge) best. In summary an inclusion of this tool into more general eLearning tools seems to be very beneficial.

11. REFERENCES

- [1] Adams, E. S., Granger, M., Goelman, D., and Ricardo, C.: Managing the introductory database course: what goes in and what comes out?. In Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2004). ACM, New York, NY, 497-498.

- [2] Bak, O.; Edmunds, M.; Nielsen, D.; Stair, N.: Web-Based Course Support Systems: An Essential Tool for Supporting Flexible Learning Environment. 19th International Conference on Database and Expert Systems Application (DEXA 2008), pp.427-430.
- [3] Brusilovsky, P., Sosnovsky, S., Yudelso, M., et al.: Learning SQL Programming with Interactive Tools: From Integration to Personalization. *Trans. Comput. Educ.* 9, 4 (Jan. 2010), 1-15.
- [4] Byrne, B., Garvey, M., Jackson, M.: Using Object Role Modelling to Teach Database Design, *Proceedings of TLAD 2003*.
- [5] Dietrich, S. W., et al. 1997. WinRDBI: A windows-based relational database educational tool. *SIGCSE'97*. 126-130.
- [6] Dietrich, S., Goelman, D.: Databases for Non-majors: The Next Challenge? (BOF session). In *Proc. of the 40th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2009)*. ACM, New York
- [7] Hillegas, R.: Teaching Databases (BOF session). In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2008)*. ACM, New York
- [8] Kearns, R., et al. 1997. A teaching system for SQL. *Australasian Computer Science Education ACSE'97*: ACM Press. 224-231.
- [9] Kenny, C., Pahl, C.: Automated tutoring for a database skills training environment. In *Proceedings of the 36th Technical Symposium on Computer Science Education (SIGCSE 2005)*. ACM, New York, 58-62.
- [10] Leimich, P., Ball, L.: Online Fun with Databases. *Proceedings of TLAD 2003*.
- [11] Meeker, R. and Nohl, D.: Using a practicum experience in your database course. *J. Comput. Small Coll.* 23, 1 (Oct. 2007), 91-96.
- [12] Mitrovic, A. NORMIT, a Web-enabled tutor for database normalization . In: Kinshuk, R. Lewis, K. Akahori, R. Kemp, T. Okamoto, L. Henderson, C-H Lee (eds) *Proc. ICCE 2002*, Auckland, 2002, pp. 1276-1280.
- [13] Mitrovic, A. 2003. An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education* vol. 13 (2003). IOS Press. 171 -195.
- [14] Monger, A.: Exploiting Effective Learning and Teaching Methods, Tools and Resources for the Distance or "Out-of-Classroom" Learning of Databases. *Proceedings of TLAD 2005*.
- [15] Murray, M. and Guimaraes, M.: Animated database courseware (ADbC): interactive instructional materials to support the teaching of database concepts. *J. Comput. Small Coll.* 24, 4 (Apr. 2009), 267-267.
- [16] Murray, M. and Guimaraes, M.: Animated database courseware: using animations to extend conceptual understanding of database concepts. *J. Comput. Small Coll.* 24, 2 (Dec. 2008), 144-150.
- [17] Pahl, C., Barrett, R., and Kenny, C.: Supporting active database learning and training through interactive multimedia. In *Proceedings of the 9th Annual SIGCSE Conference on innovation and Technology in Computer Science Education (ITiCSE 2004)*. ACM, New York, NY, 27-31
- [18] Paterson, J. H.: Teaching Databases And Objects, *Proceedings of TLAD 2008*.
- [19] Prior, J. C.: Online assessment of SQL query formulation skills. In *Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20*. Australian Computer Society, Darlinghurst, 247-256.
- [20] Ridjanovic, D.: Pedagogical tools for database design, development and use. In *Proceedings of the 2nd international Conference on Principles and Practice of Programming in Java (2003)*. PPPJ, vol. 42. Computer Science Press, New York, NY, 121-124.
- [21] Sadiq, S., Orlowska, M., Sadiq, W., and Lin, J.: SQLator: an online SQL learning workbench. In *Proceedings of the 9th Annual SIGCSE Conference on innovation and Technology in Computer Science Education (ITiCSE 2004)*. ACM, New York, NY, 223-227.
- [22] Semi annual meeting of the German Society for Computer Science's Database special interest group, Düsseldorf, November 2008, <http://herbsttreffen2008.mi.medien.fh-duesseldorf.de/programm.php>
- [23] Suraweera, P., Mitrovic, A. KERMIT: a Constraint-based Tutor for Database Modeling. In: S. Cerri, G. Gouarderes and F. Paraguacu (eds.) *Proc. 6th Int. Conf on Intelligent Tutoring Systems ITS 2002*, Biarritz, France, LCNS 2363, 377-387, 2002.
- [24] Urban, S. D., Dietrich, S. W.: *An Advanced Course in Database Systems: Beyond Relational Databases*, Prentice Hall, 2004.
- [25] Wang, J.Z.; Davis, T.; Westall, M.; Srimani, P.K.: Work in progress - MeTube: A novel way to teach database to undergraduates. 39th IEEE Frontiers in Education Conference (FIE 2009), p. 18-21.

USES OF PEER ASSESSMENT IN DATABASE TEACHING AND LEARNING

James Paterson
Glasgow Caledonian University
School of Engineering and
Computing
Glasgow G4 0BA
James.Paterson@gcu.ac.uk

John Wilson
University of Strathclyde
Dept. of Computer and Information
Sciences
Glasgow G1 1XQ
john.wilson@cis.strath.ac.uk

Petra Leimich
Abertay University
School of Computing and
Engineering Systems
Dundee DD1 1HG
p.leimich@abertay.ac.uk

ABSTRACT

This discussion paper introduces three very different methods and contexts for the use of peer assessment in introductory database classes, each of which is supported by different learning software tools. In the first case study, at Glasgow Caledonian University, Contributing Student Pedagogy is used, where students contribute to the learning of others through the collaborative creation of a bank of self-assessment questions. This is supported by the Peerwise software tool. Secondly, at Strathclyde University, students undertake formative assessment of others in providing feedback on an initial element of a larger coursework assessment. A number of virtual learning environments (VLEs) are capable of supporting this method through customisable discussion fora. Finally, at the University of Abertay Dundee, peer and self assessment are used in a group project to adjust the group grade for individual students. This is effected through the use of the WebPA software tool.

Keywords

Peer assessment; peer feedback, self assessment.

12. INTRODUCTION

Peer assessment by students of other students' work, both formative and summative, has many potential benefits to learning for all concerned. It develops the ability to evaluate and make judgements, and in doing so students gain insights into their own learning. The results of evaluation by peers can also provide a valuable source of feedback. This paper describes a variety of approaches to peer assessment and feedback, and the software tools which support them, which have been used within introductory database classes.

13. A CONTRIBUTING STUDENT PEDAGOGY IN AN INTRODUCTORY DATABASE CLASS

A Contributing Student Pedagogy (CSP) is an approach in which students contribute to the learning of others, and value the contribution of others [7]. One example of a software tool which provides support for the implementation of a CSP is PeerWise [4]. PeerWise provides a means to create an online repository of multiple choice questions (MCQs) in which students themselves write the questions. Students can then answer questions which have been contributed by others, and they have the opportunity to evaluate those contributions. The authors of PeerWise assert that asking students to write MCQs, and to provide appropriate explanations, gives a richer and deeper learning experience than simply answering practice questions which have been provided by staff [3]. The possibility that questions may be poorly thought out, or that the provided answers may be wrong, gives students an opportunity to develop skills in critical evaluation. The PeerWise system is in use in a number of institutions throughout the world. The relationship of its use to exam performance and the topic coverage represented in the students' contributions in introductory programming courses has been studied [3,5]. Recently, PeerWise has been used in an introductory database course at Glasgow Caledonian University. This section of the paper reports on the experience and on the topic coverage represented in the student contributions within that course.

13.1 Implementation

The module, Introduction to Database Development, was delivered over a short (6 week) timescale as part of a set of short introductory modules within a first year course which is common to all computing programmes. The main assessment instrument is a hand-in assignment, but there is also an online MCQ test. To encourage participation in PeerWise, a component of the overall module mark (10%) was awarded on the basis of that participation. Participation was required to be completed within weeks 2 to 5 of the module. To attain full credit, students were required to contribute at least 5 questions and answer 10 questions contributed by others. Students were made aware that they were not being assessed on the quality of their questions or the correctness of their answers. For each question, students were required to provide a question stem and a set of up to five answers, and also to indicate which answer they consider to be correct. They could optionally provide an explanation of their answer. On answering a question, a student can see the question author's choice of correct answer, and also the distribution of answers previously given for that question. It is entirely possible that the indicated correct answer may not in fact be correct, and the weight of opinion expressed in other students' answers may reflect this. The student can then optionally rate the question on a scale of 0 to 5 and provide a text comment. Factors which students may take into account in rating questions may include, for example, the correctness of the given answer and the quality of the explanation.

A total of 105 students contributed questions, which essentially is all the students who engaged with the module. Of these, only 4 contributed less than the required 5 questions. The highest number of questions contributed by any student was 10. The majority of students contributed exactly 5 questions. The total number of questions submitted was 545, and the average number of responses to each question was 2.8. Most students answered 10 questions, or a few more than that. However, 15 students answered double the required amount of questions or more, and the highest number answered by any student was 45.

13.2 Evaluation

Evaluation of the CSP approach has focused initially on two aspects. Question quality is likely to be an indicator of depth of learning. Writing a question which is challenging, and to provide good explanations for correct and incorrect choices of response requires a good understanding, as does recognizing a good question when providing ratings. Topic coverage gives a collective view of the students' viewpoint on the course material and the relative importance of each topic.

13.2.1 Question quality

Denny et al. [6] have applied a metric to measure objectively the quality of student-created questions in their courses. This has not been done yet in the initial analysis described here which focuses on the student ratings. The average rating of questions which were rated by more than 10 respondents (a figure chosen to provide a reasonable 'body of opinion') was 3.3. It is interesting to consider what students consider to be a 'good' question. For example, there was relatively little difference in average rating between a question which was a simple true/false question (True or False: SQL can be used to create a database?) and a more sophisticated question which requires a set of four SQL statements to be examined to identify the one which would correctly produce a specified result. It seems likely that experienced instructors would rate the questions significantly differently from the students. However, it is evident that students give low ratings to questions which they consider to have errors, for example where the stated correct answer is not correct, or where more than one answer could be correct. The provision of an explanation along with a question appears to have little influence on the rating. From the questions with more than 10 ratings, the average rating for questions with explanations was 3.3 compared to 3.1 for those without. In fact, less than 25% of the questions included explanations. It should be noted that no guidance was given to students on what constitutes a 'good' question. Given that the aim of the exercise is to enrich the learning experience, not to simply create a bank of practice questions, the quality of the questions is not the main concern. However, it seems likely that providing some discussion of question types and quality could be beneficial in encouraging students to devise questions which require deep understanding and to provide explanations.

13.2.2 Topic coverage

Denny et al. [4] studied the coverage of topics within student-created MCQs in an introductory programming course. They concluded that the coverage was broad, included all major topics and showed a distribution similar to that in assessments set by instructors. They also noted that a significant number of questions touched on 2 or more topics. A preliminary analysis along similar lines has been done here with the database questions. One of the decisions which has to be made in this analysis is the granularity with which topics are defined. Denny et al. used coursebook chapters as topic descriptors, and noted in support of this decision that these topics matched well to those identified in an international study of introductory programming courses

involving 200 teachers at University level [12]. We have used a similar classification according to the course structure.

The course materials were organised into 6 chapters:

1. Introduction to database systems, nomenclature and data models
2. Database design and relationships
3. SQL SELECT and aggregates
4. Normalisation
5. Indexes, SQL JOIN, INSERT, DELETE, UPDATE
6. Building database applications

This is a preliminary classification, and is not necessarily ideal. In particular, there are in some cases several identifiable topics within a particular chapter. On the other hand, the structure is used by staff to guide the setting of MCQ assessments, and is familiar to students.

The figure shows the distribution among these topics of students-contributed questions compared to that of staff-created assessment questions. Topic 7 is included to cover topics which are not explicitly included in the course but which students may have included on topics discovered through independent reading.

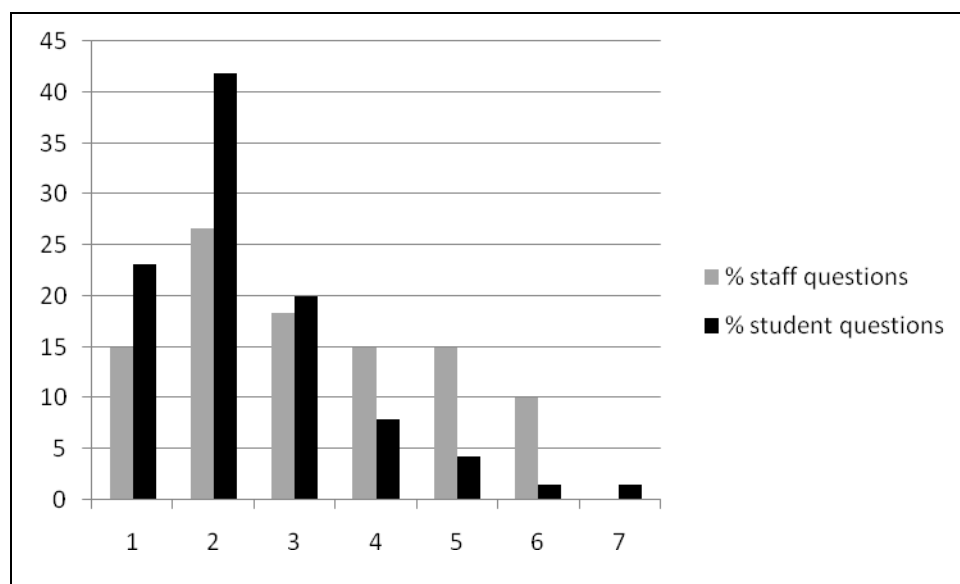


Figure 1. Topic coverage of student and staff questions. Numbers on horizontal axis refer to course material chapters, the content of which is indicated in the text

Figure 1 shows that all major topics have indeed been covered. The distributions of student and staff questions are quite different. Although both show a bias towards topic 2 (database design and relationships), the bias is more marked in student questions, while coverage of topics which are delivered later in the module is very limited. It was also noted that questions were in almost all cases focused on isolated topics, with no real integration of topics within a single question. Interestingly, the latter comment also applies to the staff questions. There may be several reasons behind these results. The timescale of the exercise was quite different from that described by Denny et al. These questions were contributed within the teaching time on a short module, rather than over a period including exam revision time after teaching, which may explain a bias towards topics taught early in the course. Students essentially had 4 weeks to contribute questions on topic 2, and less than 2 weeks for topic 5. The nature of the topics covered in a database course may lend themselves differently to creating questions than the topics in a programming course. Further work is required on topic categorisation within database courses, and on the use of PeerWise in courses with different structures and timescales.

13.3 Conclusion

The PeerWise system has been used in an introductory database course to support the creation by students of a bank of MCQs. Preliminary work has been done in analysing the contributions to investigate students' perceptions of question quality and their preference for topics on which to devise questions. For the latter, there is some evidence of differences from the picture observed in programming courses, but further study is

required to clarify this. Further study is also required to evaluate the contribution to the students' learning experience.

14. PEER-BASED FEEDBACK

The issue of feedback is consistently highlighted in the context of student opinions of their experiences in higher education. Typically universities and colleges have moved to encourage quick and constructive feedback from lecturers to their students. There is however still an issue of the need to provide intermediate feedback on students' ongoing work and it is here especially that peer-based feedback can make a particularly useful contribution. As well as helping the recipient of the feedback, students also benefit by viewing the work of their colleagues from a critical perspective. Students find that peer review based assessment encourages reflective thinking and self-improvement at the expense of additional time demands. While there are issues over the integrity of this process, simple, holistic feedback on review material submitted provides sufficient student benefit to make this approach worthwhile [10]. It is further recognised that peer based review can make a large contribution to building scholarly communities [1] as well as leading to significant improvements in the quality of student work [13].

Arranging that these benefits are available in the setting of large class cohorts is a major problem that was addressed by the Open University in the context of its introductory programming class by using a proprietary virtual learning environment. The approach developed requires four discrete stages that control the operation of a bulletin board:

1. Registration
2. First posting – bulletin boards are write only
3. Follow up posting – bulletin boards are read/write
4. Work completion – bulletin boards are read only

The stages govern the permissions on the board as shown and this fits with the work to be carried out by the student. In stage 2, students are expected to post a part of their initial solution. In stage 3, they are expected to review the work of another student and write some helpful comments. In stage 4, they complete their own work using the insight gained during stage 3 both from their own interaction and by viewing the interactions of their peers. This approach is particularly suited to complex problems where there are a number of ways of addressing the issues involved. Entity-relationship modelling presents a very apposite domain for this kind of interaction. ER modelling is essentially a social process where the designer can learn a lot from iterations in the solution space. This is enhanced by discussing the problem with others and considering alternative and sometimes incorrect viewpoints.

This study was carried out on an introductory Databases class for second level students. The class ran in a single semester and provided 10 credits. The cohort included 107 students, most of whom were registered on a BSc Honours degree in Computer Science. Students were presented with a scenario based on a hospital, which could be represented in twelve entities and eleven relationships with assorted degree and optionality. By this stage in the module, the concept of supertype/subtype entities had been introduced and there was a clear opportunity to use such a formalism in the employment hierarchy contained within the scenario. Some many-to-many relationships typically emerged during the analysis and there existed the potential for a recursive relationship to represent supervision. In all, the problem represented a variety of challenges to students and had a number of elements that were not straightforward to represent. The overall problem could generate a number of correct solutions.

A number of common software platforms are capable of supporting the previously described posting sequence. These can be categorized as virtual learning environments (VLE) or generic software that can be tailored to incorporate the necessary capabilities. The Blackboard VLE¹ provides a typical example of the former category. It supports a discussion board idiom that includes the basic functionality that is needed to control asynchronous group interaction. The system allows the instructor to manipulate fora and allocate users in particular groups to access these fora. The bulletin board supports text as well as paste-in images and attachments. Control of permissions is mainly to define post anonymity, removal and tagging of posts, thread creation and moderation. Moodle² supports a similar group-based forum concept. This implementation controls the visibility of forum contents to users from outside the forum. It restricts students to posting to one thread but permits reply postings to be made to multiple threads. Moodle allows pictures to be inserted directly into postings. Turnitin³ has also been extended to provide equivalent functionality.

¹ <http://www.blackboard.com/>

² <http://moodle.org/>

³ http://www.submit.ac.uk/static_jisc/ac_uk_index.html

WebBBS⁴ presents a typical example of the range of systems available in the second category of software platforms. It can be tailored to support the required permission pattern to structure peer-based feedback in an appropriate way. The interface is very simple with postings being categorised into threads in the same manner as Blackboard and Moodle. Images cannot be posted directly but instead can be included by writing html tags into the text of the posting.

Group 9 Discussion Board	
Phase 4: Archive of Class Work	
All Messages	
<i>12 of 12 Messages Displayed</i>	
<hr/>	
*** Welcome ***	John Wilson Mon, 19 Oct 2009, 9:55 a.m.
Enhanced ERD	Thu, 22 Oct 2009, 3:17 p.m.
Re: Enhanced ERD	Wed, 28 Oct 2009, 2:12 p.m.
Relationships	Thu, 22 Oct 2009, 11:25 p.m.
Re: Relationships	Wed, 28 Oct 2009, 4:51 p.m.
Initial Posting	Fri, 23 Oct 2009, 12:55 a.m.
Hospital in-patient phase 2 posting	Fri, 23 Oct 2009, 12:05 p.m.
Re: Hospital in-patient phase 2 posting	Tue, 27 Oct 2009, 1:38 p.m.
Phase 2	Fri, 23 Oct 2009, 3:23 p.m.
Re: Phase 2	Wed, 28 Oct 2009, 1:19 p.m.
*** End of Phase 2 ***	John Wilson Fri, 23 Oct 2009, 8:30 p.m.
*** End of Phase 3 ***	John Wilson Wed, 28 Oct 2009, 5:06 p.m.

Figure 2: Typical pattern of postings in one bulletin board group⁵

WebBBS was chosen for the forum structure because of its simplicity and adaptability. Control of the posting sequence was implemented as part of the WebBBS Perl script. Students were allocated to groups of four or five and provided with instructions on the overall objectives of the process, the details of how to make postings and the various deadlines. Figure 2 illustrates the typical sequence of interactions between users of the system. A good posting will include a description of a difficult part of the problem and comments such as:

"In the above diagram I have produced the relationship treat between patient and doctor. However, in the scenario it says that a patient is assigned to a consultant. What I am unsure of is whether my diagram should be changed so that the relationship is between patient and consultant"

A poorer solution (eg Figure 3) would simply list all the entities and relationships in the scenario without an attempt to draw out the difficult elements. In this particular case, the student also experienced difficulty with the process involved in posting images to the bulletin board. The follow-up messages (shown indented in Figure 2) are posted to the prior contributor, apart from the post to the last contributor, which was posted by the first contributor in the group. They vary in scope from a few terse comments to a more helpful and expansive explanation (Figure 4)

14.1 Conclusion

This approach to organizing peer review has been found to be very effective in motivating student involvement in assessed coursework well in advance of the final deadline. The imposition of intermediate deadlines ensures that work is carried out in a more even manner than is typically the case with a single deadline. Most students take part in the process and make contributions during both the posting phases. The overall quality of the postings is variable, with some students misinterpreting the instructions and posting their complete solution

⁴ <http://awsd.com/scripts/webbbs/>

⁵ Postings are not anonymous although identifying details have been removed from this Figure.

or making only a very terse contribution. Similarly response postings vary considerably in quality. Students are instructed that the tone of the response postings is to be constructively critical and this always maintained. The postings are not anonymous and this may produce a constraint on the material that is posted. Whilst WebBBS provides a useful platform for this kind of interaction, the approach to posting images can present technical challenges to students at this level.

Group 9 Discussion Board

Phase 2

Posted By:

Date: Fri, 23 Oct 2009, 3:23 p.m.

Databases – Phase 2

Entities:

Nurse, Ward, Bed.

Consultants, Junior Doctors, Drugs, Patients

Consultants, Drugs, Patients

List of Relationships:

Nurses, Ward - Many:1 - Mandatory

Consultants, Patients - 1:Many - Mandatory

Junior Doctors, Patients - Many:Many - Mandatory

Nurses, Patients - Many:Many - Mandatory

Consultants, Drugs - Many:Many - Optional

I have completed the entity relation diagram but I'm not sure how it would be uploaded.

Figure 3: Typical poor quality posting

Group 20 Discussion Board

Re: Databases

Posted By:

Date: Wed, 28 Oct 2009, 2:18 p.m.

In Response To: [Databases](#)

Hi, here are my responses to the questions you raised:

"I am not sure about the relationships between drugs and patients, and between drugs and doctors."

From the descriptions of the relationships that you've posted I can see that you're thinking about using M:N relationships between patient/doctor and drug/doctor. M:N relationships are to be avoided and in order to do so I'd suggest creating an intermediate entity that exists between doctor, patient and drug. If you do this you'll end up with simpler relationships between the original entities and this new entity instead.

"Do doctors always prescribe a drug for a patient?"

It strikes me as though a doctor does not have to prescribe a drug to a patient. It does not explicitly mention this in the description we were given but in real life plenty of people are admitted to hospital and

Figure 4: Typical good quality response posting

15. PEER ASSESSMENT IN THE ALLOCATION OF GRADES FOR GROUP PROJECTS

The introductory database module at Abertay is taken in semester 2 of the first year by students from a number of computing-related courses. The class contact comprises a one hour lecture and two 1.5-hour labs per week per student, over 12 weeks. Group projects are used as the major assessment. These have been successful in enhancing and sustaining students' interest in the subject matter, and working closely with peers generally helps students undertake more active enquiry, contributing to the group effort. Also, reserving part of the weekly lab sessions for project work enables a close dialogue between the students and lecturer. The final group product is graded, and forms the majority of the module grade. As with group projects in general,

students are concerned that grades should be fair, i.e. vary depending on the contributions made by individuals. This is mirrored by staff concerns that students should not pass the module purely on the strength of others. Group presentations, where students are graded individually, contribute to this but are insufficient as the weighting is fairly low compared to the product itself. Therefore the grading of the product (the group's database application and associated documentation) also required a way of individually adjusting grades. Initially, an attendance-based method was used. Students were allowed to miss two of the weekly in-lab group meetings without penalty, to account for unavoidable illnesses etc. However, the grades of students who missed more than two meetings were reduced by one number grade (on a 20 point scale where 20 represents A+, 9 represents D-, 0 non-submission) per meeting missed. This method was found to address the issue of contribution, but to have some major drawbacks: firstly, it relies on the lecturer maintaining accurate attendance records at all times, secondly, and more importantly, it also penalised students who made their contribution to the group work mainly outside of class, while not penalising students who contributed little despite attending class. Finally, this method had no student input, which is now regarded as good practice in assessment and evaluation (see, e.g. [11]). In an attempt to improve the process, and to avoid the drawbacks described above, peer assessment as part of the grading process was first introduced last year.

15.1 Implementation

In researching the experiences of others, several themes emerged. Several authors have described problems where students were required to discuss and agree their grades publicly within the group. For example, Cogdell et al [2] reported that students "did not like giving low marks to colleagues face to face. Consequently non-contributors would get the same marks as everyone else and the rest of the group would feel resentful. Alternatively the group would mark a member down and this person would complain vociferously." Other authors agree that peer assessment should be performed in private; for example, Lejk and Wyvill [9] found that students were more discriminating in their peer assessment when it is performed secretly. Lejk and Wyvill [9], among others, also emphasise the importance of including self-assessment in the process, in order to avoid over-generous students effectively penalising themselves. Kennedy [8] presents another, similar scenario, and found that there was little overall variation introduced in the grades through the peer assessment process, and that many students expressed reluctance to judge their peers. On the other hand, Kennedy observed that other students were keen to discriminate, and that this could lead to dysfunctional groups, with uneven distributions of tasks in the group right from the beginning of a project, when domineering students ensured they undertook the most demanding and credit bearing tasks. Kennedy questions the reliability and validity of the process also because of observed wide inconsistencies in students' judgment of each other. In Kennedy's scenario, self assessment was not incorporated, and whether the peer assessment was public or confidential is not stated.

Based on the literature, it was decided that the peer assessment used should be confidential, and include self-assessment. WebPA [14, 15], open source software developed for this setting by the University of Loughborough, was used to facilitate the process and minimise any administrative burden. The students in each group were asked to rate every group member in five distinct areas, based on those suggested by WebPA [14]:

1. Co-Operation: This covers attendance at meetings, contribution to meetings, carrying out of designated tasks, dealing with problems,. helping others in the group.
2. Communication: This covers effectiveness in meetings, clarity of work submitted to the group, negotiation with the group, communication between meetings and providing feedback about the contributions made by others in the group.
3. Enthusiasm: This covers motivation, creativity and initiative during the project, including finding out about methods beyond the taught materials.
4. Organisation: This covers skills in self-organisation and the ability to organise others. It also covers planning, setting targets, establishing ground rules and keeping to deadlines.
5. Contribution: This covers the overall effort put in by an individual during the Semester.

For each area, a rating scale of 0-5 was used:

Score 0 : no help at all

Score 1 : quite poor

Score 2 : not as good as most of the group

Score 3 : about average for this group

Score 4 : better than most of the group

Score 5 : really excellent.

	A	B	C	D	E	F	G	H	I	J	K
2	Student A	GM 0	GM 1	GM 2	GM 3	GM 4		Group mark overall		80%	B16
3	Q1 : 1. CO-OPERATION	4	5	4	3	3		Web-PA score			
4	Q2 : 2. COMMUNICATION	4	3	5	2	1		Student A	1.16	86.52%	B17
5	Q3 : 3. ENTHUSIASM	4	4	4	3	3		Student B	1.4	95.99%	A19
6	Q4 : 4. ORGANISATION	3	3	3	2	2		Student C	0.98	79.06%	B16
7	Q5 : 5. CONTRIBUTION	4	5	4	1	2		Student D	0.5	60.11%	C12
8								Student E	0.96	78.32%	B16
9	Student B	GM 0	GM 1	GM 2	GM 3	GM 4					
10	Q1 : 1. CO-OPERATION	5	4	5	3	2					
11	Q2 : 2. COMMUNICATION	5	4	5	1	1					
12	Q3 : 3. ENTHUSIASM	5	5	5	3	3					
13	Q4 : 4. ORGANISATION	4	4	4	4	1					
14	Q5 : 5. CONTRIBUTION	5	5	5	5	5					
15											
16	Student C	GM 0	GM 1	GM 2	GM 3	GM 4					
17	Q1 : 1. CO-OPERATION	4	5	3	3	0					
18	Q2 : 2. COMMUNICATION	4	4	4	2	0					
19	Q3 : 3. ENTHUSIASM	4	4	3	3	0					
20	Q4 : 4. ORGANISATION	3	5	3	2	2					
21	Q5 : 5. CONTRIBUTION	4	5	3	1	0					
22											
23	Student D	GM 0	GM 1	GM 2	GM 3	GM 4					
24	Q1 : 1. CO-OPERATION	0	0	1	3	2					
25	Q2 : 2. COMMUNICATION	0	0	2	3	1					
26	Q3 : 3. ENTHUSIASM	0	0	1	3	3					
27	Q4 : 4. ORGANISATION	0	0	0	3	2					
28	Q5 : 5. CONTRIBUTION	1	0	1	3	3					
29											
30	Student E	GM 0	GM 1	GM 2	GM 3	GM 4					
31	Q1 : 1. CO-OPERATION	2	1	3	3	2					
32	Q2 : 2. COMMUNICATION	1	1	4	4	2					
33	Q3 : 3. ENTHUSIASM	2	2	4	3	4					
34	Q4 : 4. ORGANISATION	0	1	3	4	2					
35	Q5 : 5. CONTRIBUTION	2	2	4	5	4					

Figure 5. Sample WebPA output for one project group

The WebPA software allows staff to review individual results and also aggregates the scores into a final grade. Staff also select the weight of the peer assessment component, in this case 50%. An example of the WebPA output for a single project group, is shown in Figure 5. The grades in column K are on the Abertay grading scale (where A20 is the best possible; D9 the lowest pass mark). The group mark, B16, was converted to a percentage and then the individual grades converted back to the Abertay scale.

15.2 Evaluation

Once installed by Information Services, WebPA was easy to set up. This requires a number of steps, uploading module and student information (using templates provided), creating the assessment with relevant dates and a marking form, assigning the students to groups. WebPA allows for the use of single sign-on. In the second year of operation however, unforeseen problems were encountered in that most students could not log in and received no error message either. Eventually, the problem was fixed by Information Services. It turned out that authentication relied on the students' email address, rather than their user name, and due to very recent migration of the student email provider, email addresses did not all follow a single pattern. Despite these teething problems, which resulted in a two-week delay and countless unsuccessful attempts, students were keen to participate. However, it was decided not to apply the 100% non-submission penalty for students who had clearly participated in the project and earlier plans to seek formal feedback on the process from students this session were abandoned.

Once the students had left their peer and self assessments, group grades were entered into WebPA and its algorithms applied to calculate individual grades. The results, and detailed ratings, were inspected closely by the lecturer. As expected from the observations of group work in labs, several patterns of group outcomes emerged. In some groups, there was very little (or even no) variance, resulting in all students being awarded

the group grade. In other groups, there were large differences, resulting in a wide differentiation of individual grades.

As in the example of Figure 5 above, the five different categories were used effectively, and different scores given. This indicates that students took care in arriving at their assessments. Many added thoughtful and reflective comments to their scores. This helped reconcile the very occasional wide differences between the scores given to one individual by the group members. The written comments were also useful in the few cases where the ratings were very different from the lecturer's personal impression formed during classes. Where necessary, academic judgment and performance in other module elements were used to arrive at a final grade. There was no evidence of students "ganging up" on a group member, or of students trying to improve their grade through giving themselves a very high score.

Informal feedback was received from several students by email. This showed that students found the software easy to use. One student was initially concerned: "Can I just confirm that my ratings for the other group members will not be displayed for all to see? I just feel that this will cause problems as already there have been accusations of 'Back stabbing' between group members, it's crazy I know." Following assurance that individual ratings were confidential, the student commented: "It worked fine and was very easy to use! I think all group work modules should have this at the end, it's excellent."

15.3 Conclusion

While a full evaluation is yet to be carried out, initial experiences with this method based on two instances of operation are positive. The system has now been used with two cohorts, each with about 20 groups and 80 to 100 students. Not a single complaint has been received about the system's use or the resulting grade, indicating that students do find the system fair.

16. OVERALL CONCLUSION

The three very different examples introduced in this paper illustrate the potential for the use of peer assessment and feedback in introductory database classes in a wide variety of contexts and illustrate a range of software tools which are available to facilitate these processes. The approaches and tools used have been shown to be applicable in the teaching and learning of databases and have succeeded in engaging students in peer activity. Further work is required to evaluate the impact on learning and to identify activities and topics within database classes which may benefit from these approaches. As an aside, while the use of these tools is not specific to databases, or any other subject for that matter, they may have one additional use which is unique within our discipline: one thing they all have in common is that they each make use of a database and they could be nice examples of case studies which demonstrate the real-world importance of databases!

17. ACKNOWLEDGEMENT

PeerWise was created at and is hosted by the University of Auckland, New Zealand, and the use of this facility is greatly appreciated. Thanks especially to Paul Denny for his support and advice.

18. REFERENCES

- [1] Chang, C.; Chen, G & Li, L., Constructing a community of practice to improve coursework activity, *Computers & Education*, **50**(1) 235-247 (2008).
- [2] Cogdell, B; Brown A & Campbell A., Peer-assessment of group work in a large class – development of a staff and student friendly system. In: Orsmond, P. *Self- and Peer-assessment. Guidance on Practice in the Biosciences*. 35-37. Available at <ftp://www.bioscience.heacademy.ac.uk/TeachingGuides/fulltext.pdf> (2004)
- [3] Denny, P.; Hamer, J.; Luxton-Reilly, A. & Purchase, H., PeerWise: students sharing their multiple choice questions, *ICER '08: Proceeding of the Fourth international Workshop on Computing Education Research*, ACM, New York, NY, USA, 51-58 (2008).
- [4] Denny, P.; Luxton-Reilly, A. & Hamer, J., The PeerWise system of student contributed assessment questions, *ACE '08: Proceedings of the tenth conference on Australasian computing education*, ACS, Darlinghurst, Australia, 69-74 (2008).

- [5] Denny, P.; Luxton-Reilly, A.; Hamer, J. & Purchase, H., Coverage of course topics in a student generated MCQ repository, *ITiCSE '09: Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education*, ACM, New York, NY, USA, 11-15 (2009).
- [6] Denny, P.; Luxton-Reilly, A. & Simon, B., Quality of Student Contributed Questions Using PeerWise, *Proceedings of the Eleventh Australasian Computing Education Conference*, ACS, Wellington, New Zealand, 45-53 (2009).
- [7] Hamer, J., Some experiences with the "contributing student approach", *ITiCSE '06: Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education*, ACM, New York, NY, USA, 68-72 (2006).
- [8] Kennedy, G.J., Peer-assessment in Group Projects: Is It Worth It? *Proceedings of the Seventh Australasian Computing Education Conference*, ACS Newcastle, Australia (2005).
- [9] Lejk, M. & Wyvill, M., The Effect of the Inclusion of Self assessment with Peer Assessment of Contributions to a Group Project: a quantitative study of secret and agreed assessments. *Assessment and Evaluation in Higher Education*, **26**(6), 551-561 (2001).
- [10] Lin, S.; Liu, E. & Yuan, S., Web-based peer assessment: feedback for students with various thinking-styles, *Journal of Computer Assisted Learning*, **17**(4), 420-432 (2001).
- [11] Nicol, D & Macfarlane-Dick, D. Formative assessment and self-regulated learning: a model and seven principles of good feedback practice. *Studies in Higher Education* **31**(2), 199-218 (2006)
- [12] Schulte, C. & Bennedsen, J. What do teachers teach in introductory programming? *ICER '06: Proceedings of the second international workshop on Computing education research*, ACM, New York, NY, USA, 17-28 (2006).
- [13] Sung, Y.; Chang, K.; Chiou, S. & Hou, H., The design and application of a web-based self- and peer-assessment system, *Computers & Education*, **45**(2), 187-202 (2005).
- [14] Tenant, J.; Crawford, A. & Wilkinson, N., Supplementary Information About Self and Peer Assessment. Available at <http://staffcentral.bton.ac.uk/clt/events/2007conf/documents/9%20wilkinson%20%20bates%20supp%20info.doc>
- [15] WebPA. Loughborough University. Available at <http://www.webpaproject.com/>.

SQL PATTERNS A NEW APPROACH FOR TEACHING SQL

Huda Al-Shuaily
University of Glasgow
University Avenue, Glasgow G12 8 8RZ, United Kingdom
huda@dcs.gla.ac.uk
<http://www.dcs.gla.ac.uk/~huda>

Karen Renaud
University of Glasgow
University Avenue, Glasgow G12 8RZ, United Kingdom
karen@dcs.gla.ac.uk
<http://www.dcs.gla.ac.uk/~karen>

ABSTRACT

Skill in querying databases using SQL is a fundamental outcome expected by industry from anyone who has completed a course in Database systems. Unfortunately students often have difficulty with applying fundamental SQL concepts and find writing SQL queries a complex task. This paper proposes a new approach for teaching SQL by using SQL patterns which are based on a checklist approach. Patterns and pattern languages are widely considered to be a useful tool and method to model design experience, both in architecture, where they were originally conceived, as well as in many computer-related fields. Many researchers have proved that using patterns can help the students to learn the intended concepts. This paper introduces the use of SQL patterns in database education and proposes their use in training students to write complex queries in SQL.

Keywords

SQL, PATTERN, CHECKLIST, TEACHING.

19. INTRODUCTION

Structured Query Language (SQL) is today the standard language for querying relational and object-relational databases. Constructing database queries in (SQL) is a pivotal skill required by many developers because of the interaction of application programs and databases. Therefore, learning SQL has become mandatory for all Computer Science students. There are several aspects of SQL that cause difficulties. Some researchers attribute this to the nature of SQL that it is fundamentally different from the other skills that students have to master [14].

This paper discusses some of the issues and proposes using patterns to solve the highlighted issues. The pattern approach is used as a structure to present the common scenarios and appropriate common solutions. We argue that understanding SQL design patterns is critical to SQL learner so that learners understand the common SQL design patterns and have a framework for common SQL solutions. SQL is unlike traditional programming languages. There are no mechanisms for flow control, loops, variables and no methods for storing intermediate results [7]. Hence “SQL leverage ‘predicates’ a logical structure that is perfect for design patterns” [7].

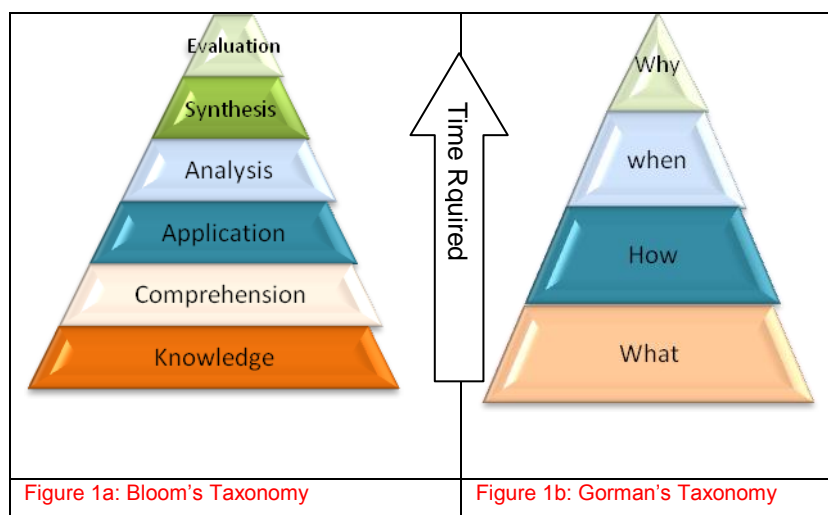
The structure of the SQL patterns which are introduced here are unlike other design patterns. They are written based on the checklist approach [11, 16, and 17] which will reduce the complexity of the patterns and make it easier for students to match the pattern to the problem.

20. TEACHING SQL

Before we start discussing the methods of teaching SQL and the problems encountered within those approaches, we need to make sure that we understand the nature of learning SQL. In fact, students learn SQL by solving problems and reflecting on their experiences; an approach known as Problem Based Learning (PBL) [18]. In this section we will describe the learning taxonomy, PBL approach, and how they relate to learning SQL. Then we will discuss some current issues in solving SQL queries that students face during their assignments and exams. Looking at how humans accumulate knowledge make the problem in learning SQL much clearer. In the beginning, let us discuss two taxonomies: Bloom and Gorman. Bloom (1956) proposed a taxonomy which classified forms of learning. He identified three major levels of learning, and argued that upper levels should not be attempted before lower levels had been mastered. Bloom’s Taxonomy consists of

six stages: knowledge, comprehension, application, analysis, syntheses and evaluation [23], as shown in Figure 1a. Gorman (2002) proposed a simplified taxonomy, as shown in Figure 1.b, with just four levels: What,

How, When and Why [24]. Comparing both taxonomy, we can find that Gorman's "What" aligns with Bloom's "knowledge and comprehension" while Gorman's "How" aligns with Bloom's "application level". Furthermore, Gorman's "When" aligns with Bloom's "Analysis". Finally Gorman's top level "Why" aligns with Bloom's "Evaluation".



Renaud et al. looked at theories of learning such as those of Bloom and Gorman and noted that the reason students sometimes have difficulty applying database skills, such as SQL, is that they do not get the required knowledge. The authors argue that it is thus very important to convey core knowledge first and later students can construct skills (such as SQL) up on top of that core knowledge. [22]. In other words, we are teaching the knowledge (what) but not how, when or why to apply in a certain context. Students learn the concepts during the lectures and learn how to apply by solving many problems. In today SQL courses, students experience many difficulties in matching the knowledge learnt in lectures with that knowledge to the given SQL problems in the lab. Because, students do not know how to apply such knowledge, when to apply or why to apply it. That is a result of their not having experience in solving SQL problems. The question is how to build such an experience within one or two courses. We are arguing that students need to be exposed to SQL problems and to solve them. Then, students can build a strong background in solving SQL queries. However, students are consuming a lot of time and efforts solving each problem depending on its complexity. We need to understand why students are consuming such time and effort, and why they sometimes give up solving questions in their assignments and exams. As we mentioned, students learn SQL by solving problems using Problem Based Learning (PBL) [18]. Figure 2 shows the problem-based learning cycle.

This cycle, also known as the PBL tutorial process, begins when the students are confronted with a scenario that represents the problem. Their immediate action is to identify the relevant facts which help them to represent and understand the problem better. This leads to the generation of a hypothesis about different solutions. Occasionally, there are gaps in the knowledge that requires identification and research through self-directed learning (SDL). The students will apply this new knowledge in relation to their problem statement and the hypothesis and will eventually evaluate this situation to end up with an abstract knowledge that they need to reflect upon. Figure 2 illustrates this cycle [18].

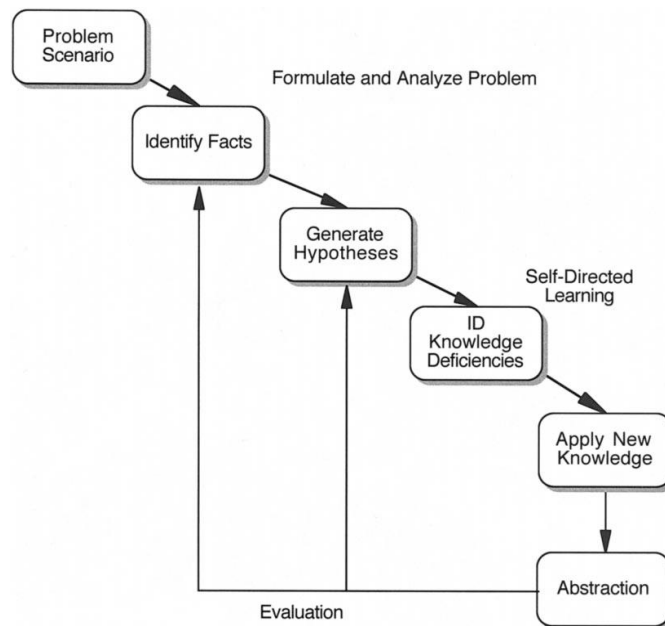


Figure 2: The problem-based learning cycle [18]

To clarify this further we need to discuss the steps that students follow in solving SQL problem and highlight the main issues in each step. Also, we will discuss how our approach will help in minimizing such issue:

Step1: Fact Identification (The When):

When students are given a problem and asked to solve it by writing SQL queries, they are supposed to analyze the problem and try to understand the main facts that are embedded. However, a student's level of knowledge and experience in solving such a problem will be the major driver in performing this step successfully. Here are some questions that we would like to discuss at this step:

- [1] Are students able to identify the relevant facts from the scenario? Students need to understand the given problem and be able to list related facts depending on how complex the problem is.
- [2] How long does it take them to identify these facts? This depends on both problem complexity and students' knowledge.
- [3] How accurate are these facts? Sometimes students misinterpret the query at this stage, and when the foundation is incorrect it leads to an incorrect query.

According to the previous learning taxonomies, students need to get some knowledge about SQL concepts (What) and the way to apply such knowledge to a given problem (How) before attempting to solve any SQL query.

Step 2: Idea Generation (The How):

After analyzing the problem, the students need to decide how to solve the problem by generating some ideas and hypotheses. However, let us look at these questions:

- [1] Are they able to generate hypotheses and ideas? The ideas that will be generated will be driven by the fact identification step. Renaud and Biljon state that 'SQL is essentially a declarative language, which allows us to specify what we want and not how to get it' [14]. Hence, even when students are able to identify the facts of the problem (what), they have difficulties identifying (how) to solve it.
- [2] How close are these hypotheses and ideas to the given problem? SQL writers have to work with sets and reason about values [14]. As students become experts in solving SQL queries, their ability to solve complex query become better.

Step 3: Knowledge deficiencies (The What):

Are the students able to identify holes in their own knowledge to implement their Ideas? This is not necessarily a given. Actually, this is probably where most difficulties occur. Bratvold, Begg and Campbell [12] argue that many people, especially those with the least knowledge, overestimate their own performance and ability. They find that those who most need training are the least likely to acknowledge it. In learning, this probably means that novice SQL writers will not necessarily be aware of the deficiencies in their knowledge. Bjork [20] agrees

that most people who fail to assess their competence accurately in a given area will err on the side of overconfidence. Kruger and Dunning [21] argue that students' lack of competence in a given area will rob them of their ability to realize it and to take remedial action. Thus this particular phase is where we have to focus our attention – and this is why we present design patterns here. If learners are able to identify the facts and generate ideas, then the design patterns will provide the lacking knowledge in a convenient format. This does not rely on the learner's own assessment of their knowledge, which might well be completely wrong: the required knowledge is simply provided in a handy format for the learner to use. In addition, in terms of time spent to find related information, Mitrovic notes that "unrestricted exploration is not advisable, especially for novices, as students may waste too much time wandering"[25]

Pausing to reflect on the first three steps which we considered as the majors facts in solving SQL problems, we can realize where the major issues are.

- [1] Firstly, students do not have enough knowledge analyzing any SQL problem which makes it more difficult.
- [2] Secondly, exposing students to SQL problems where the concept is newly introduced and not yet mastered by the students prevent the students from finding the facts that are embedded within the SQL problem or even in generating ideas to solve it.
- [3] Thirdly, novice SQL writers will not be aware of the deficiencies in their knowledge and also consider the time they will take to find the required knowledge.

Providing SQL learner with SQL design patterns will help students to become familiar with common SQL problems and related solutions. Therefore, student's knowledge and experience will be enhanced. In addition, SQL Patterns provide the lacking knowledge in a convenient format. This does not rely on the learner's own assessment of their knowledge, which might well be completely wrong: the required knowledge is simply provided in a handy format for the learner to use.

Step 4: Applying new knowledge (The How):

During this stage, students use self-directed learning (SDL) to apply their hypotheses. Renaud and Biljon conclude that SQL is a skill and learning it is "essentially constructive" and therefore we "need to identify the foundation" upon which SQL is built [14]. Furthermore, "the absence or insufficient understanding of these concepts will inevitably lead to failure to truly understand SQL". The previous stage, if done correctly, will lead into this stage. By the time the learner progresses to this stage, he or she should have the wherewithal to apply the new knowledge.

Step 5: Abstract knowledge:

At the completion of each problem, the students reflect on the abstract knowledge gained.

Step 6: Evaluation - Write and Test the Query (The Why):

Students will eventually need to evaluate their hypotheses in light of what they have learned. They apply the knowledge and assess the results here. During the evaluation stage students might revisit the first (fact identification) or the second (idea generation) steps.

Studying the learning taxonomy and the PBL (figure 3a,3b and 3c) one can notice that students attempt to perform the upper level of the learning taxonomy before having mastered the knowledge encapsulated in the lower levels. That's why students have a shaky foundation, which leads to poor results in learning SQL. Therefore, we are proposing an approach that is intended to bridge the gap between what students are doing in solving SQL queries (which is based on PBL) and what learning theory tells us about how people learn. Anderson argues that discovery-based learning leads to greater retention of knowledge, which is obviously what PBL is striving towards [26]. However, in terms of learning theory, this discovery-based approach could lead to frustration and the student giving up. The point of the pattern is to bridge the gap and to guide the discovery process. This will prevent the student from wasting a great deal of time searching for answers in the wrong places. Whereas exploration of the available information is good if this activity is productive [25], but unguided exploration could just as easily lead to students becoming discouraged and not learning anything at the end of the day. How the patterns will help students to solve SQL problems by avoiding the discussed issues. This will be discussed further in section4. The example shown in the next section demonstrates how students should solve a simple SQL problem.

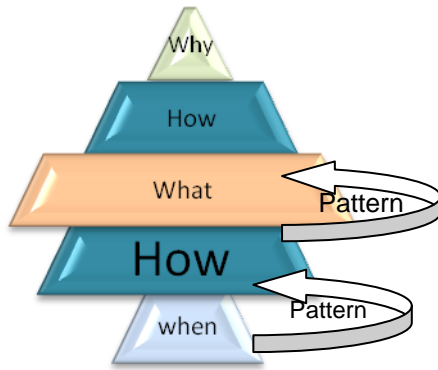


Figure 3a: How students are expected to solve SQL problems in light of problem-based learning in terms of learning theory. The arrows depict the role of SQL patterns

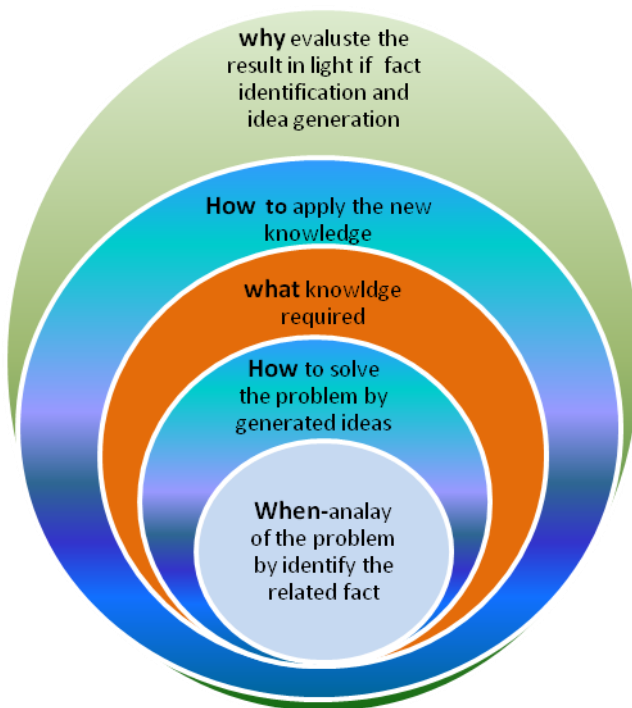


Figure 3b: How students solve SQL problems in light of problem-based learning in terms of learning theory. The circle depicts the amount of time and effort spent at each stage.

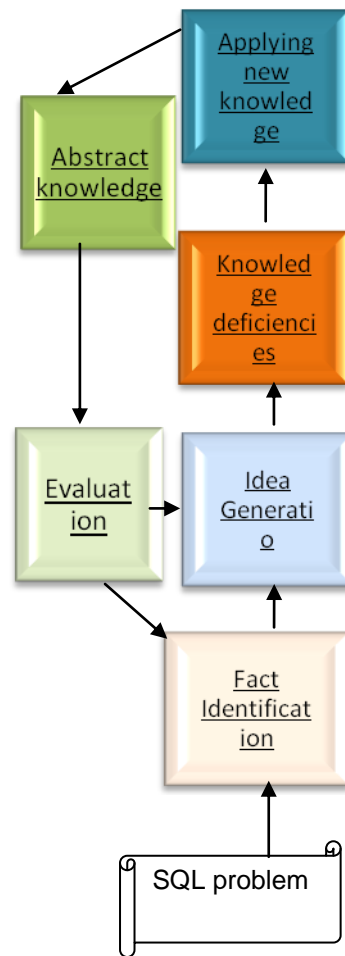


Figure 3b: Solving SQL problem under problem-based learning (PBL) theory

20.1 An Example

To be precise, when students are attempting a particular SQL problem scenario, they should follow the following steps. Let us look at this scenario:

Example 1: Write a query to display Employee Name, Department name and Salary for each employee that earning salary between 500 and 1500

Fact Identification:

- [1] Details of all employees must be displayed
- [2] Details are not all in the same table

Idea Generation:

- [1] Gather related information from multiple tables
- [2] Tables need to be joined by matching values in related columns. Need to select the required matching columns from the two tables i.e. Those which should match to ensure that the data in one table is related to the data in the other.
- [3] Not all details need to be returned by the query

Knowledge deficiencies (Syntax):

- [1] Understand the correct terminology for this action i.e. join
- [2] Determine the correct SQL syntax to gather information from multiple tables i.e. name both tables in FROM, and use WHERE to specify which column values should be matched
- [3] Finding out how to filter details from joined tables i.e. specify column names in SELECT

New knowledge:

```
Select e.Emp_name, d.Dept_name, e.Salary
From Employee e, Department d
Where e.Dept_Id = d.Dept_Id
And salary between 500 and 1500;
```

There are a few points that we need to address regarding the above scenario:

- [1] This scenario is a common query that is carried out daily in organizations.
- [2] The facts that are identified will apply to all similar problems.
- [3] Therefore, the generated ideas should match the facts.
- [4] The knowledge deficiencies should provide the bridge between ideas and implementation. Without this the learner might well get stuck after the idea generation.
- [5] Most similar queries use similar code (with a change in name of the tables and columns), which makes the pattern applicable in various contexts.

If this is the case, then, our approach is to collect all similar problems and their related solutions so that it can be used by others (perhaps a novice learner). This is ultimately what we will refer to as a 'SQL pattern'. In other words, each generic problem type, with its related facts, generated ideas and the required knowledge will be collected together to become a 'pattern'.

To solve the highlighted issues in teaching and learning SQL we propose a new technique based on two main approaches: patterns and checklists. As mentioned earlier, the pattern approach is used as a structure to present the common scenarios and appropriate common solutions. The checklist approach is used to ease identification of the pattern which matches the common problem. Section 4 will elaborate the pattern concept further. However, before doing that we will discuss, in section 3, how to write each problem related fact and generated idea using the 'checklist approach'.

21. CHECKLIST APPROACH

A checklist can be described as "a list of factors, properties, aspects, components, criteria, tasks, or dimensions, the presence, referent, or amount of which are to be considered separately, in order to perform a certain task" [16]. There are different types of checklists, as defined by Scriven [16]. For example: sequential checklist, strongly sequential checklist, weakly sequential checklist, diagnostic checklist and the criteria of merit checklist. We are using diagnostic checklist.

Atul Gawande followed this same approach and obtained some interesting results which he recounts in his latest book, The Checklist Manifesto: How to Get Things Right. He argues strongly that checklists were an effective remedy to "ignorance, uncertainty and complexity" [11].

21.1 Value of Checklists in Learning SQL

When students are given a complex task, they are up against three main difficulties: faulty memory, distraction [11] and poor assessment of their competence. In addition, some query writers skip crucial steps even when they remember them. The checklist approach provides protection against such failure [11]. Rowlands recommends using checklists to support student learning and performance by suggesting that well-designed checklists identify steps students should take to complete complex tasks [17]. Strickland provides students with checklists to "catalogue the items that should be included in a project or task"[19].

In this paper, we will focus our discussion on checklists that support students in solving SQL queries. As was discussed in section 2, students need to derive the related facts from the given SQL problem. Many students cannot list all the facts because they simply don't understand what is being asked. Providing students with similar SQL problems written as a checklist might well make it easier for them to match and select the related fact.

22. SQL PATTERNS

The idea of patterns emerged from the architect Christopher Alexander [9] and is now commonly used to systematize the main principles and pragmatics in the Architectural fields. Those ideas have inspired many other fields like IT and education to use patterns. Therefore, within the computer science field, different types of patterns appear for example Software Engineering Patterns, HCI patterns, SQL design patterns and pedagogical patterns. There is a growing interest in the possibility of using patterns in teaching.

SQL patterns are just like any other patterns in one respect. In the same way that anyone can apply standard code design patterns in programming languages, he/she can also apply design patterns to SQL. SQL design patterns have appeared in a book written by an Oracle employee, Vadim Tropashko [7]. His SQL patterns apply directly to Oracle SQL. This manual is valuable as it categorizes and describes the most common SQL structures and design patterns. It would be beneficial for an expert as they must understand the most efficient way of writing SQL for complex database queries. Managers may insist on formulating these processes as mandatory in the production and maintenance of their organization's databases so as to improve these standards and therefore improve the quality and productivity of system development projects. This book is unique in its approach and serves the professional as well as the academic since it is founded on theory suitable for all types of SQL query problems. However, the novice learner cannot utilize these patterns because of their limited knowledge and experience in writing SQL.

Therefore we are introducing a new set of SQL patterns that are intended to help the novice learner to master SQL in a limited time. We are firm believers in creating symmetry through design patterns where it is applicable. Using good modelling techniques and naming conventions on data and process definitions will enable easier code generation. All patterns have a systematic structure like the one below.

Pattern section	Definition
Reference	This part will have a number, so each pattern will contain a unique number. This is used to link or refer different patterns
Name	Each pattern will have a name that is easy to remember and track
Keyword	A few words that summarize the content of the pattern. These keywords will be used for later search about any patterns when the collection is in electronic status.
Problem	SQL common problems will be presented here
Fact identification	A checklist of the problem related facts will be presented here
Idea generation	The solution will be based on checklist approach where a number of scenarios will be listed and the learner will select the most appropriate.
Knowledge required	Many code structures will be presented. Each will match one or more scenario that was selected on previous section (solution)
Examples	This will provide SQL code and table snapshot that refer to above code structure showing a step-by-step display of how the result of any query can be calculated. The reason of this is to use visual presentation and to animate the execution of the code so that students can develop better mental models of what is described.

Table 1: pattern's structure

Let us look at this scenario: when students are given the SQL question and SQL pattern in table 1. In the SQL problem, students have to retrieve some information that resides in two tables and it should meet some criteria. Then, we are assuming that students can notice two things: that they need to harvest information from two tables and link them to "Select" and "Join" in SQL knowledge. This is found in the Keyword section within

each pattern. Many patterns can be found that include “Select” and “Join”. Now students can look at the problem section where they can match the problem in their hand with the problem in the pattern. As soon as students can match the right pattern with the given problem, they will be able to solve the problem. The other sections: Fact identification, Idea generation, Knowledge required and the Example will provide them with the required knowledge to write the query. Table 2 shows another pattern.

Reference	0001																								
Name	Querying from multiple tables																								
Keywords	SELECT, JOIN																								
Problem	Gather information from two tables.																								
Fact identification	<div>[] Information is stored in more than one table</div> <div>[] The rows need to be filtered (OPTIONAL)</div> <div>[] The returned columns need to be filtered (OPTIONAL)</div>																								
Idea generation	<div>Need to link related rows in the two tables</div> <div>Need to filter only the rows that are required by the query</div> <div>Need to filter only the columns that are required by the query</div>																								
Knowledge required	<div>1. Linking two tables is referred to as <u>joining</u> them. This is done by identifying a column in each table which is used as the link between them. In SQL, this is formulated as: WHERE table1.column_name = table2.column_name</div> <div>2. We filter <u>rows</u> by using a conditional statement in the WHERE part of the query. E.g. WHERE gender="Female"</div> <div>3. We filter <u>columns</u> in the SELECT part of the query. E.g. SELECT Name</div>																								
Example	<div>Get the names, department names and salaries of employees whose salary between 500 and 1500.</div> <div>Facts: Information is stored in Employee and Department tables. Both Rows and Columns need to be filtered</div> <div>Ideas: Need to link the two tables using department_id</div> <div><table><tr><th>Dept_Id</th><th>Dept_name</th><th>Loc_Id</th></tr><tr><td>10</td><td>Accounting</td><td>0011</td></tr><tr><td>20</td><td>HR</td><td>0013</td></tr><tr><td>30</td><td>IT</td><td>0101</td></tr><tr><td>40</td><td>Sale</td><td>0015</td></tr><tr><td>50</td><td>Shipping</td><td>0018</td></tr><tr><td>60</td><td>Marketing</td><td>1101</td></tr><tr><td>70</td><td>Services</td><td>1105</td></tr></table></div>	Dept_Id	Dept_name	Loc_Id	10	Accounting	0011	20	HR	0013	30	IT	0101	40	Sale	0015	50	Shipping	0018	60	Marketing	1101	70	Services	1105
Dept_Id	Dept_name	Loc_Id																							
10	Accounting	0011																							
20	HR	0013																							
30	IT	0101																							
40	Sale	0015																							
50	Shipping	0018																							
60	Marketing	1101																							
70	Services	1105																							

	<table><tr><th>Emp_name</th><th>Dept_name</th><th>Salary</th></tr><tr><td>Ali</td><td>Shipping</td><td>1500</td></tr><tr><td>Fay</td><td>Shipping</td><td>1300</td></tr><tr><td>Ross</td><td>Services</td><td>700</td></tr></table> <p><i>SELECT e.Emp_name, d.Dept_name, e.Salary</i> <i>FROM Employee e, Department d</i> <i>WHERE e.Dept_Id = d.Dept_ID</i> <i>And Salary Between 500 and 1500;</i></p>	Emp_name	Dept_name	Salary	Ali	Shipping	1500	Fay	Shipping	1300	Ross	Services	700	<table><tr><th>Emp_Id</th><th>Emp_name</th><th>Dept_Id</th><th>Salary</th></tr><tr><td></td><td>e</td><td></td><td></td></tr><tr><td>113</td><td>Ali</td><td>50</td><td>1500</td></tr><tr><td>205</td><td>Fay</td><td>50</td><td>1300</td></tr><tr><td>206</td><td>Ross</td><td>70</td><td>700</td></tr><tr><td>101</td><td>Ahmed</td><td>20</td><td>2000</td></tr><tr><td>100</td><td>King</td><td>20</td><td>5000</td></tr></table>	Emp_Id	Emp_name	Dept_Id	Salary		e			113	Ali	50	1500	205	Fay	50	1300	206	Ross	70	700	101	Ahmed	20	2000	100	King	20	5000
Emp_name	Dept_name	Salary																																								
Ali	Shipping	1500																																								
Fay	Shipping	1300																																								
Ross	Services	700																																								
Emp_Id	Emp_name	Dept_Id	Salary																																							
	e																																									
113	Ali	50	1500																																							
205	Fay	50	1300																																							
206	Ross	70	700																																							
101	Ahmed	20	2000																																							
100	King	20	5000																																							

Table 2: Pattern - Querying from multiple tables

Reference	0002
Name	Using Subqueries
Keyword	Subquery
Problem	Gather information
Fact identification	<p>[] Report information from one table – referred to as MAIN table</p> <p>[] Filter the information based on data in another table – referred to as SECONDARY table</p> <p>[] The returned MAIN table columns need to be filtered (OPTIONAL)</p>
Idea generation	<p>Describe the result needed from the secondary table</p> <p>Decide how to use that result to filter the main table's data</p> <p>Need to filter only the columns that are required by the query</p>
Knowledge required	<ol style="list-style-type: none"> The query on the secondary table is called a subquery or inner query. It is usually enclosed in brackets in the outer query <p>OUTER QUERY</p> <p>(INNER QUERY)</p> The inner query returns a SET of values, and these values are used in the WHERE section of the outer query to filter rows in the main table. Eg. <p>SELECT *</p> <p>FROM main</p> <p>WHERE somevalue in</p> <p>(select values from secondary)</p> If the inner query returns only one value, we could use: <p>SELECT *</p> <p>FROM main</p> <p>WHERE values =</p> <p>(select values from secondary)</p> In number (3) above, the outer query checks for values IN the set returned by the inner query. The outer query can also check for the existence (or non-existence) of returned values. Eg. <p>SELECT *</p> <p>FROM main</p>

	<div><div>WHERE EXISTS</div><div>(select values from secondary where someconstraint)</div><div>Or</div><div>SELECT *</div><div>FROM main</div><div>WHERE NOT EXISTS</div><div>(select values from secondary where someconstraint)</div></div>																								
Examples	<div><div>Get the name of the person who earns the lowest salary.</div><div>Facts: Main table is employee, secondary table is employee. We need only the name of the person who earns the lowest salary.</div><div>Ideas:</div><div><div>1. Inner query needs to return one value: the lowest salary.</div><div>2. Outer query needs to use this result to filter rows of employee table to return only the employee whose salary matches the lowest salary returned by the inner query.</div></div><div><div><div>Main query</div><div><div>Subquery</div><div>returns</div><div>single row</div></div></div><div><div><div>SELECT Emp_name</div><div>FROM Employees</div><div>WHERE Salary =</div><div>(SELECT MIN(Salary)</div><div>FROM</div><div>Employees);</div><div>Return</div><div>Return</div><div>700</div><div>Ross</div></div></div><div><table><tr><th>Emp_Id</th><th>Emp_name</th><th>Dept_Id</th><th>Salary</th></tr><tr><td>113</td><td>Ali</td><td>50</td><td>1500</td></tr><tr><td>205</td><td>Fay</td><td>50</td><td>1300</td></tr><tr><td>206</td><td>Ross</td><td>70</td><td>700</td></tr><tr><td>101</td><td>Ahmed</td><td>20</td><td>2000</td></tr><tr><td>100</td><td>King</td><td>20</td><td>5000</td></tr></table></div></div></div>	Emp_Id	Emp_name	Dept_Id	Salary	113	Ali	50	1500	205	Fay	50	1300	206	Ross	70	700	101	Ahmed	20	2000	100	King	20	5000
Emp_Id	Emp_name	Dept_Id	Salary																						
113	Ali	50	1500																						
205	Fay	50	1300																						
206	Ross	70	700																						
101	Ahmed	20	2000																						
100	King	20	5000																						

Table 3: Pattern of single row subquery

23. CASE STUDY

The research described here focuses on the application of SQL patterns in the process of solving a complex query. A design case offers a realistic framework for exploring, using and observing the usability of SQL pattern in practice. The context of research was solving SQL problem using SQL patterns. The task was to solve the given problem first without SQL patterns, and then using some relevant SQL patterns. The following SQL problem was given:

Write an SQL statement to find all employees who earn more than the average salary in their department. Display Last name, Salary, Department Id and the Average salary for the department. Sort by Average salary

They were provided with the following patterns in the second stage of the task:

- "Group Function" pattern
- "Grouping Rows" pattern
- "Sub Queries" pattern.
- "Querying from one table twice" pattern

The study results were analyzed in light of a pre task and post task questionnaire. A pre-task questionnaire was given to the participants to provide insight on their level of knowledge and experience in SQL. Three PhD

students participated. All consider themselves novice SQL developers. One student had worked with SQL before. Two students had 0-6 months and 1 had more than 3-5 years experience in working with SQL. The participant's solutions (with/without patterns) were analyzed as follows: skills in exploring the problem and identifying the related facts, the correctness of the SQL query, and the ability to match the given patterns to the given SQL problem.

In our study we found that most of the participants could not solve the problem without the given patterns as all claimed that it was hard to remember how to solve the query and all they could remember was the select statement. All of them agreed that SQL patterns helped them to recall their knowledge and provided them with the core SQL constructs they required to solve the given problem. However, most of the participants could not produce a 100% correct solution. Common participant errors include missing the linkage clause from self-join table query and they did not include the non-aggregated attributes in the GROUP BY clause although the given patterns included such information.

The main contribution lies in the fact that this case study investigates the usefulness of SQL patterns from the participant's point of view and at the same time how correct the participant's solution is. The current study is too small to be conclusive, but what emerges is that a more substantial study is required to confirm the value of SQL patterns in helping the novice to solve more complex queries. The study is considered an indirect assessment of the conventional approach in teaching SQL patterns.

24. CONCLUSION AND FUTURE WORK

In this paper we have discussed some of the issues in teaching and learning SQL; mainly in solving SQL problem where novice struggle in providing the right solution and we have proposed SQL patterns as a solution to overcome some of the discussed issues. A preliminary study of using SQL patterns in solving SQL problem was presented and conclude that future research could investigate the contribution of SQL patterns in novice learning SQL as well as SQL developer. The study aims to focus on developing skills in solving complex query as well as the effort (time, correctness, etc) taken by learner. Furthermore, this study highlights on the users comprehension of the main advantage of using SQL patterns. We intend on embarking on further research in order to refine SQL patterns and to obtain more empirical evidence of its efficacy in learning SQL.

25. REFERENCES

- [1] Kearns, R., Sheard, S., & Fekete, A.: A teaching system for SQL. In: Proceedings of the second Australasian conference on Computer Science education, Melbourne, Australia (1997) 224–231
- [2] Goldberg, C.: Do you know SQL? About Semantic Errors in Database Queries. In: 7th Workshop on Teaching, Learning and Assessment in Databases, Birmingham, UK, HEA (2009) (to be published)
- [3] Brass, S., Goldberg, C.: Semantic Errors in SQL Queries: A Quite Complete List. *Journal of Systems and Software* 79(5) (2006)
- [4] Mitrovic, A. Learning SQL with a computerized tutor. In proceedings of SIGCSE'98,(1998), p307-311.
- [5] Ramsden, P. Learning to teach in higher education. London, Routledge, 1992.
- [6] Scott. Best Practices for SQL Design Patterns (2006). Retrieved March 5, 2010, from <http://dataglass.blogspot.com/2006/02/best-practices-for-sql-design-patterns.html>
- [7] Tropashko, V. SQL Design patterns the professional guide to SQL programming. Rampant Techpress, 2007.
- [8] Faroult, S. The Art of SQL. O'Reilly Media Inc. 2006.
- [9] Alexander, C. (1979). The timeless way of building. Oxford, UK. Oxford University Press.
- [10] The Value of Checklist. A blog Websit. <http://aboveandbeyondkm.com/2010/01/the-value-of-checklists.html>
- [11] Gawande, A. The checklist manifesto, How to get things right. New York. Metropolitan Books. 2009.
- [12] Bratvold, R. B., Begg, S.H., Campbell, J.M., 2002, Would you know a good decision if you saw one? : Society of Petroleum Engineers (SPE) paper 77509.
- [13] Dadashzadeh, M. Teaching tip: A simpler approach to set comparison queries in SQL. *Journal of Information Systems Education*, Vol. 14(4) (2003)
- [14] Renaud, K. and Biljon, J. Teaching SQL - Which pedagogical Horse for this course? BNCOD 2004, LNCS 3112, pp. 224-256, Springer, Heidelberg (2004)

- [15] Matos, V. and Grasser, R. Teaching tip: A simpler (and better) SQL Approach to Relational Division. *Journal of Information Systems Education*, Vol. 13(2) (2002)
- [16] Scriven, M. The logic and methodology of checklists (December 2007). Retrieved March 10, 2010, from: http://www.wmich.edu/evalctr/checklists/papers/logic&methodology_dec07.pdf
- [17] Rowlands, K. Check It Out! Using Checklist to Support Student Learning. *English Journal*, Vol. 96, No.6 (July 2007)
- [18] Hmelo-Silver, C. E. Problem-based learning: What and how do students learn? (2004) *Educational Psychology Review*, 16, 235–266.
- [19] Strickland, K. Making assessment elementary. Portsmouth, NH: Heinemann (2000).
- [20] Bjork, R . Assessing our own Competence: Heuristics and Illusions. Chapter 15 in *Attention and Performance: Cognitive Regulation of Performance: Interaction of Theory and Application 17th: Symposium Proceedings (Attention & Performance) (Hardcover)* by D Gopher MIT Press (11 May 1999)
- [21] Kruger, J, Dunning, D. Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology*. Vol 77(6), Dec 1999, 1121-1134.
- [22] Renaud, K., Al-Shuaily, H., Cooper, R.: Facilitating Efficacious Transfer of Database Knowledge and Skills. In: 7th Workshop on Teaching, Learning and Assessment in Databases, Birmingham, UK, HEA (2009) (to be published)
- [23] Bloom B S (ed.), *Taxonomy of Educational Objectives, the classification of educational goals – Handbook I: Cognitive Domain* New York: McKay (1956).
- [24] Gorman, M. E., Types of knowledge and their roles in technology transfer. *Journal of Technology Transfer*. Volume 27, Number 3. pp219-231 (2002).
- [25] Mitrovic, A. A knowledge-based teaching system for SQL. In T. Ottmann & I. Tomek (Eds.), *Proceedings of ED-MEDIA '98* (pp. 1027-1032). Charlottesville, VA: AACE. (1998).
- [26] Anderson, J. R, Corbett, A. T., Koedinger, K. R., Pelletier, R. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4(2), 167-207, (1995).

A RELATIONAL ALGEBRA TUTOR

David Nelson

Faculty of Applied Sciences,
University of Sunderland,
Sunderland, SR6 0DD, UK
david.nelson@sunderland.ac.uk

Jonathan Scott

Faculty of Applied Sciences,
University of Sunderland,
Sunderland, SR6 0DD, UK
bd77cn@student.sunderland.ac.uk

ABSTRACT

Relational algebra is a fundamental aspect of relational database theory that it is important for students to understand. Experience suggests that students find it difficult to grasp the main concepts for a number of reasons. One of these is due to the lack of practical implementations which would enable students to test their understanding of the algebra. A small number of practical implementations of relational algebra do exist, for example the WinRDBI teaching tool [1] and the db++ DBMS [2], however these are primarily text based and therefore can be error prone and unintuitive.

In this presentation we will demonstrate and discuss a graphical relational algebra teaching tool which has been developed to be used by undergraduate and postgraduate students who are studying database modules. This project is the result of a final year undergraduate project, and the presentation will outline the need for a relational algebra teaching tool, give a demonstration of the tool, and an evaluation. The web-based interface uses drag-and-drop functionality, to minimize errors and increase usability, and the presentation will discuss student's experiences of using this type of interface for developing relational algebra queries. The tool also assists the learner's understanding of relational algebra by displaying the SQL equivalent to their relational algebra query, as well as the result of the query, so that the students are able to evaluate the correctness of the queries that they have produced as well as improving their understanding of the relationship between the algebra and SQL.

An evaluation was carried out on a Master's cohort of students and the preliminary results from that evaluation will be discussed. In the test half of the students were given the system to use to answer a number of queries, and half were required to produce their answers on paper. The evaluation will therefore compare the student's experiences of using the interface as a learning tool as against less sophisticated means of learning relational algebra. The students were asked to provide feedback on using the tool and their feedback will be evaluated, in particular a number of suggestions for improvement to the tool were given, some of which are leading to further student projects.

Keywords

Relational algebra, relational theory, teaching tool.

26. REFERENCES

[26] WinRDBI Educational Tool, <http://winrdbi.asu.edu/>

[27] Concept ASA Software + Consulting GmbH, http://www.concept-asa.de/index_gb.html

MAKING THE MOST OF AN E-LEARNING PLATFORM TO PROMOTE COLLABORATIVE LEARNING: A CASE STUDY

Liz Sokolowski
Thames Valley University
St Marys Road
London
liz.sokolowski@tvu.ac.uk

ABSTRACT

This paper describes a group work assignment forming part of the end of module assessment on a second year undergraduate database design and management course at Thames Valley University. Students were put into teams of four and allocated a case study which was used for the analysis, design and implementation of an appropriate relational database. The group work element of the assessment involved each team member taking on the role of Lead Researcher to investigate and then instruct their peers on a specific database management issue that had been introduced in a lecture, but required further analysis and application to the case study. The group work was conducted through group facilities set up on the University's VLE. The combined efforts of each group were assessed by means of a presentation given to tutors for which a single group mark was given. Lead researchers were encouraged to pitch their findings at an appropriate level, to critique each other's work and to devise ways of transferring their learning to their peers. This promoted increased levels of collaboration and engagement, which enhanced the learning process and resulted in high levels of achievement overall. The use of the VLE ensured transparency in the group working process and provided a more convenient and effective way for students to collaborate.

Keywords

Collaborative learning; group work; student centred learning

27. BACKGROUND

Database Design and Management (DDM) is a core second year module running on all Computing UG courses at Thames Valley University (TVU). The University has an overall commitment to broadening diversity and widening participation in higher education which manifests itself in a diverse student population with a range of abilities and backgrounds. Many students come from ethnic minorities or are international students where English is not their first language. The DDM module aims to cover the main principles and techniques involved in designing, implementing and managing relational databases and is delivered on all computing related full time and part time courses.

28. STRUCTURE AND DELIVERY OF THE MODULE

The first part of the DDM module focuses on database analysis and design techniques and on gaining practical skills in SQL. Students are introduced to some preliminary entity relationship modelling and MS Access database in a first year module and these basic concepts are developed further in DDM using progressively more difficult examples. Oracle is used to explore the capabilities of SQL and to construct a small but non trivial relational database for one of a number of case studies provided by the tutor. The second half of the module focuses on an introduction to some of the issues involved in managing databases, including database security, transaction management, performance monitoring and optimization issues. The full time undergraduate course is delivered as a one hour lecture and double practical session run over a single semester spanning 15 weeks.

29. USE OF THE VLE IN DATABASE TEACHING

Over the years increasing use has been made of Blackboard, TVU's Virtual Learning Environment (VLE), to support teaching and learning, though this has tended to be at level 2 (Document Distribution) within the five level model proposed by Van der Craats, McGovern and Pannan (2002) [1], with lecture slides, supporting handouts and assessments being made available online. Students were also given a fairly extensive SQL

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

tutorial containing SQL exercises to work through in preparation for the first assessment, a written SQL test. The basic problem encountered with this approach was that many students took the materials and disappeared, reappearing only for the test which was attempted with varying degrees of success.

As a result of this, it was decided to introduce short weekly online multiple choice tests. When used appropriately, it has been shown that class tests can improve attendance and provide effective formative feedback to students on the progress of their learning [2]. They also reflect basic good practice of reflecting on previous material in courses where skill or knowledge is built up progressively [3]. The mini tests enhance student engagement in the material and prepare them for a more extensive SQL paper based test which students sit in week 6. In order to motivate students, the best three out of four tests contribute 10% to the total module mark. Students are also encouraged to 'fox your peers'. This involves formulating challenging query requirements, which, if unsolved by other members in the class, attract extra marks.

The use of this teaching strategy and these assessment methods has resulted in high pass rates for the practical SQL element of the course. The purpose of the intensive SQL element is to prepare students for the end of module assessment which runs in three parts. The first involves constructing a data model with associated documentation for a case study provided by the tutor. Four to five different case studies are used so that each is allocated to up to four students, randomly selected, who then make up a team with responsibility for producing a working system for that scenario using Oracle isqlplus. No user interface is implemented at this stage. Discussions are carried out between members to discuss scope and assumptions, but the actual implementation is done as an individual assignment. The final part of the assessment involves group work to produce a 30 minute presentation on various database administration aspects of the case study.

30. ASSESSMENT USING GROUP WORK

The group work element of the assessment has been subjected to numerous revisions over time, as problems have been encountered and reacted to. The choice of group work has clear and well documented benefits for this type of task: as well as reflecting the real world where individuals are required to cooperate with others to achieve a task it also allows students to learn from each other and for the group to undertake more substantial investigations than they might have undertaken on their own. Another increasingly important consideration is that growing student numbers have created pressure for more efficient methods of teaching and assessing students. Marking group assignments is one way of reducing the volume of marking and managing increasing numbers [4].

However, group dynamics are often difficult to manage and group work difficult to assess. Group work on the module was subject to the classic and well documented problems associated with team working: the unfairness of the lazy student who contributes little but ends up well rewarded thanks to the efforts of his peers; or, worse still, the deserving student who fails a module having been encumbered by a dysfunctional group on a crucial group work assessment [4]. A number of different options were tried to deal with these issues, each with their merits and drawbacks. The radical solution would have been to eliminate group work altogether and ask students to write individual reports on the topic areas. However, the result of this would inevitably be more superficial coverage of the issues. The learning objectives required students to apply some fairly challenging concepts introduced in lectures to their case study organization and it was felt that that time and the constraints of TVU guidelines on the length of assessments for second year modules would preclude any meaningful investigation of these topic areas. Moreover, some of these issues are not straight cut and require a more discursive approach.

31. PROBLEMS WITH ASSESSING GROUP WORK

In order to deal with issues of differing levels of effort from students, an element of peer assessment was initially introduced. This allowed group members to redistribute a portion of the pool of marks awarded by a tutor for the end of module presentation, thus rewarding industrious members with higher marks and reducing marks for members who hadn't pulled their weight. The scheme was discussed and agreed with students. Yet when given this chance to alter the marks of errant group members, very few students grasped this opportunity. In fact it was noticeable that students who were often the most vociferous when it came to complaining about their group members, failed to make any adjustments to individual scores. Whether scoring was done in secret or in the open made no difference in this respect. This was a common problem in peer assessment. Students tended to see this as the job of the tutor, not the student. The view appeared to be why should they have to suffer the wrath of a friend whose mark they had been responsible for downgrading?

Another issue concerned the question of what exactly students were marking. Was it the process employed in producing the final product, or was it the quality of the product itself? Should a student be rewarded for turning up on time to all meetings, or should that student be rewarded only if he/she actually contributed something

meaningful? This was a source of conflict between group members. Students who had missed the occasional meeting argued that they had contributed more than other students who had attended regularly. However, levels of contribution to product were difficult to assess, whereas contribution to process was easily appraised by checking minutes of meetings and this therefore tended to be used as the basis for allocating marks. Ideally these two elements should both be present in any group work assessment [5]

32. THE SOLUTION ADOPTED

In view of the above problems, it was decided to take assessment away from students and back into the hands of the tutors and to utilize Blackboard to provide transparency of both process and product. Each group of four students was allocated a group facility on Blackboard, with a discussion board, group email, file download area and virtual classroom permitting online chat. Groups were private, i.e. inaccessible to anyone outside the membership of the group, with the exception of the tutor who had guest access.

Each group member was asked to take on the formal role of Lead Researcher for one of the four topics to be covered in the final presentation: database security, integrity, performance and backup. Their topic choice had to be agreed with the rest of the team within a week of the assignment being issued. The lead researchers were expected to investigate and publish initial findings on their group's Blackboard facility by a deadline set by the tutor. The initial findings could be in the form of a short presentation or an overview document, supported by a paper or Internet article which they considered to be particularly relevant. Other members of the group had then to comment on the clarity and relevance of the initial findings, using the appropriate discussion thread on Blackboard. A deadline was also set for feedback from group members. In this way every member critiqued the work of the others and evidence of this was provided on the group's discussion board or file transfer area. The lead researcher was responsible for taking the feedback on board and for producing PowerPoint slides for his/her topic for inclusion in the final presentation.

Students were reminded that all members of the team would be questioned on all aspects of the presentation, not just the areas they were responsible for developing, so lead researchers were tasked with devising methods for ensuring that other team members would be confident to field questions on their topic area. Tutors monitored Group pages on a weekly basis and provided generic feedback or advice, as necessary, in the form of an announcement posted to the whole cohort.

A presentation lasting 20 minutes was given by each group in the final week of the course, with each lead researcher presenting his/her topic. This was followed by 10 minutes of questions. The group work was assessed by two tutors who awarded marks based on:

- Scholarship and understanding;
- Application of findings to the case study allocated to the group;
- Ability of group members to answer questions on the material presented;
- Quality of the presentation delivery.

33. EVALUATION

7.1 The Tutor Perspective

The scheme described above was implemented as an experiment in cooperative student driven learning involving two seminar groups each with around 20 students. The overall results were encouraging and it was deemed a success overall, though with some reservations.

To start with, all but two of the students adhered to the strict timescales set down by tutors. Initial research was posted in good time, but although approximately half the cohort had made a serious attempt to evaluate and summarise findings from various sources, the other half simply 'cut and paste' large quantities of material from the Internet and passed this off as their initial research. In response to this, an 'all user' announcement was posted to the VLE by a tutor reminding lead researchers to check that their postings were in a form that could be easily digested by the rest of the team. A more powerful impetus for change came, however, from the more pointed messages from other students in the group who complained in no uncertain terms about the quality of the material and asked for it to be amended.

Feedback from team members was also variable, with most students diligently taking on their role of reviewer (Figure 1), while others contented themselves with a few cursory comments (Figure 2).

Home Page Courses My Links Library Services Student Support Communities Content Collection Open

Here is my feedback on your notes so far. Overall good stuff

X Presentation Notes

1. Make it more specific to Company: More detail on data ie customers, employees, sales etc
2. Would all data be secured in the same way ? Would there be a priority for specific data areas
3. Data threats could be badly trained data entry staff. Have data quality protocols guide ?
4. Read only access for staff in warehouse. Need to print address labels from dbase for boxes
5. Accounts- Deal with suppliers. Sales/Credit Control deal with Customers
6. Passwords-Strength monitors need to be alphanumeric 6-8 characters change every 3 months
7. Security Policy update every 6 months as matter of course be aware of emerging threats

Y Presentation Notes

1. Maybe mention number users and affect of busy periods eg 9am-11am
2. What are expected queries and transactions?
3. Set performance parameters. What do you expect from system?, back end and front end eg time to retrieve a record.
4. Mention Oracle tools that solve problems eg ADDM (Automatic Database Diagnostic Monitors) Oracle.com is a good pl to look for more info.
5. Indexing- Rapid access to certain area,s improves performance. 3 types B Tree Bitmap and Function

Don't forget to look at alternative sources and quote them if necessary

Figure 1: Example of comprehensive initial feedback from group member to lead researchers

Total views: 5 Your views: 1

Hi

I have read everyone's research notes and i think its ok to put into the presentation slides.

Figure 2: Example of short response to initial feedback from group member to lead researchers

The minimum expectation was that there should be some acknowledgement that students had considered the material that had been posted by their team members. Lead researchers were instructed to evaluate all the feedback they received, remembering that they were the 'experts' in their area. They were encouraged to adopt a critical approach to what was put before them, rather than blind acceptance of other students' suggestions. The aim was to foster students' confidence in their capacity as learners, as beliefs about this have been shown to affect achievement [6].

The next question posed to lead researchers was "what means are you going use to ensure that your group members understand and can answer questions on your topic?" The solutions were varied: references to face to face meetings, online briefings and even a multiple choice test could be followed in the discussion threads on Blackboard.

Tutors monitored the group pages periodically and also posted occasional hints or general observations in the form of announcements on the VLE, but they did not interfere in or influence group processes directly. This was left to the students. One of the main points needing to be reinforced was that good grades would not be achieved unless the material was clearly applied to the group's case study. For example, when looking at security, it was not enough to talk in general terms about restricting access to data; the presentation had to address the specific roles, access rights, views, etc that would need to be set up for the group's case study database.

Contact between team members was very largely online, right up until a day or two before the assessment date, when groups needed to come together to rehearse their presentations.

Judged overall, the content of the presentations was marginally better than that of previous years; the main difference however was a marked improvement in student cooperation, confidence and ability to answer questions. It is difficult to explain why this was so, after all, the process followed online mirrored in general terms what normally happens with group work: work is split up between students, students go away to research a one aspect of the work and then come back to assemble the results into a coherent assignment. It is the tutors' belief that in the case of the DBMS experiment, evidence of group work activity in terms of both its process and product could be tracked on each group's Blackboard site and this was instrumental in ensuring that students produced the deliverables expected of them. Secondly, the esteem of some of the weakest students from difficult backgrounds was enhanced by their lead researcher status which allowed them to produce initial findings without being pressurised or influenced by stronger members of the team.

Analysis of the discussion threads showed that in most cases there had been a serious attempt at evaluation and synthesis of material covered in the assignment and at passing on knowledge to fellow team members. Significantly, with the exception of two students who failed to post initial research, a single mark was awarded to all group members with no complaints from students.

7.2 The Student Perspective

Student satisfaction and motivation are important factors in measuring the success of any new teaching and learning strategy. An evaluation questionnaire completed by students revealed positive feedback on the group work. 94% of students "strongly agreed" or "agreed" that Blackboard group facilities had been helpful in supporting the group work; 75% "strongly agreed" or "agreed" with the statement that "the group work allowed me to learn a lot from other group members" (the remainder were neutral); and 80% "strongly agreed" or "agreed" that all group members contributed in equal measure to the group work (with two members "strongly disagreeing" and the rest neutral). The last two questions revealed a significantly higher satisfaction rate than in previous years where responses in the agreeing categories tended to hover around the 50% mark. However, a notable statistic was that in spite of this favourable feedback, 54% of students agreed with the statement that "I would have preferred to write a report on all four topics covered in the presentation, rather than do group work", testimony perhaps to the difficult nature of group work.

34. GENERAL FINDINGS AND CONCLUSION

Much attention has been focused in recent years on constructivism and alignment in education [7] and on the importance of the reflective practitioner in striving to achieve this in designing learning outcomes and assessments. This paper describes changes made to the teaching of the DBMS module in order to improve the group work experience of students so that the learning objectives of the module could be better achieved. The changes harnessed the capabilities of a VLE to provide an environment for a more effective and transparent group working process, showing that when groups work well, students learn more [8]. Although group work may not be the assessment of choice for many students, it reflects the reality of the work environment and develops experience and skills that cannot easily be acquired by other means and for this reason merits the attention of reflective practitioners.

35. REFERENCES

- [28] Van der Craats C., McGovern J. & Pannan L.. A five-level approach to the large-scale development and delivery of on-line programs (2002)
<http://www.ascilite.org.au/conferences/auckland02/proceedings/papers/182.pdf>
- [29] Haigh M. Sustaining learning through assessment: an evaluation of the value of a weekly class quiz. *Assessment & Evaluation in Higher Education* **32**:4, 457-474 (2007).
- [30] Ramsden P. *Learning to Teach in Higher Education*. Routledge (1992).
- [31] Nordberg D. Group Projects: More Learning? Less Fair? A Conundrum in Assessing Postgraduate Business Education. *Assessment & Evaluation in Higher Education* **33**:5, 481-492 (2008)
- [32] Humphreys P., Greenan K. and McIlveen H. Developing work-based transferable skills in a university environment. *Journal of European Industrial Training* **21**:2, 63-69. (1997)
- [33] Boud D. and Falchikov N., Aligning Assessment with Long-term Learning. *Assessment & Evaluation in Higher Education* **31**:4, 399-413 (2006).
- [34] Biggs J., Enhancing Teaching Through Constructive Alignment. *Higher Education* **32**:3, 347-364
- [35] Mills B. J. and Cottell, P.G. *Cooperative Learning for Higher Education*. American Council on Education & Oryx Press, Westport (1998)

AN ENTITY-RELATIONSHIP DIAGRAM (ERD) MARKING SCHEME

Paul Henderson
Department of Computing
Sheffield Hallam University
Sheffield S1 1WB
p.l.henderson@shu.ac.uk

ABSTRACT

The Marking of Entity-Relationship Diagrams (ERDs), where the student is required to use top-down modelling, is often challenging. Lecturers use their judgement when awarding marks, but the precise nature of such judgements can be hard to define. This workshop introduces an approach to this problem which incorporates ideas to reduce judgement whilst marking, creating greater consistency across multiple variations of solutions whilst speeding up the process of marking. This approach is embodied in a spreadsheet.

Keywords

Spreadsheet, Marking, Entity-Relationship Diagrams.

36. RATIONALE

When marking ERDs that have been produced top-down, there are no precursors to assess the ERD by, as in the case of normalisation, for example. The ERD has to be assessed on its own and, however many possible variations on the correct solution the lecturer can envisage, students always seem to think of more.

A moderately-sized ERD can yield many incorrect solution permutations when students freely add more entities or relationships, misname entities and misplace foreign keys. In these situations, there is a temptation to develop some marking criteria on the fly as the number of permutations increases, which can lead to the situation where subsequently the lecturer may not remember how marks were arrived at in all cases. This can cause problems during moderation or when students question their marks.

It was felt that there may be some aspects of judgement which could be pre-defined and incorporated into spreadsheet formulae, such that simple data entry could account for an element of the ERD marks.

The aim of the spreadsheet was not to remove judgement completely (even if that were possible) rather to pre-define some aspects of the solution to any ERD in order that the lecturer need not waste time in some of the repetitive judgemental activities which could result in marking inconsistencies in those aspects of the model. This results in a semi-mechanistic process, where running judgement is still needed, but some of the judgements are pre-defined in spreadsheet formulae.

37. DEVELOPMENT

The spreadsheet has undergone a number of iterations over several years, where the spreadsheet marking results have been compared to what the author or colleagues would have awarded manually and formulae have been modified to more accurately reflect the manual mark across a range of scripts. Conversely the spreadsheet results have also challenged colleagues to justify their manual marks.

The spreadsheet offers a number of tuning facilities which can be set empirically to produce the desired result.

38. WORKSHOP FORMAT

The workshop format will be as follows:

- 20 minutes introduction and demonstration of teaching material
- 15 minutes participation by delegates
- 15 minutes discussion and feedback.

To run this workshop the use of computer lab by delegates will be required.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

TEACHING UNDERGRADUATE STUDENTS DATA MINING: IDEAS, EXPERIENCE AND CHALLENGES

Hongbo Du
Department of Applied Computing
University of Buckingham
Buckingham MK18 1EG
hongbo.du@buckingham.ac.uk

ABSTRACT

Data mining has become an important subject in computing. Teaching the subject to undergraduate students at level 6 has its own difficulties and faces with unique challenges. This paper is a reflection of the author's experience in teaching the subject at both undergraduate and postgraduate levels over many years. The paper first presents an algorithm-oriented syllabus that follows a white-box approach to data mining and then attempts to address a number of issues in relation to learning, teaching and assessment of this complex subject. The paper briefly summarises the author's experience in delivering the subject in different places and lessons learnt. The paper is intended to initiate a discussion over the best practice, the expectation and the relationships of data mining with other relevant subjects in a degree programme.

Keywords

Data mining, undergraduate, syllabus, learning, teaching, assessment.

39. INTRODUCTION

In recent years, data mining has become a popular and interesting subject in computing. The technology has a wide range of potential applications, and has therefore attracted a great deal of attention from organizations of various kinds. After nearly two decades of extensive research and practice, the knowledge of the subject has sufficiently matured to be down-streamed to classrooms. Indeed, attempts have been made recently to introduce the subject to master students. The number of specialist postgraduate programmes in data mining in UK universities is increasing ([6], [9], [10]). However, teaching the subject as a stand-alone module at the undergraduate level is still relatively rare [11]. Instead, in many places where data mining is taught at all, the subject is briefly covered as a topic within an advanced database course [8]. The time may have finally come to add this interesting subject to the curriculum in computing related disciplines.

The author is among a small group of lecturers in teaching data mining primarily to an undergraduate audience since the early days of data mining. This paper is a reflection on his extensive experience in teaching the subject in different places over the years. Teaching data mining to undergraduate students is not an easy task. There is little agreement among course tutors regarding how the subject should be taught and what topics should be covered. This short article is an attempt to address the related issues.

The paper first presents a course syllabus that is primarily designed for undergraduate students from software engineering, computer science, computer systems and applied computing. The paper argues that a *white-box* algorithmic approach is the most suitable way of teaching data mining techniques and methods to this audience. At the same time, the paper stresses the importance of striking a balance between concepts, theory and practice. The paper attempts to address issues in relation to learning, teaching and assessment of the subject, and summarise some useful lessons learnt from the author's experience in the past. At the end, the paper is intended to initiate a discussion over the best practice and expectation in the teaching of this complex subject, and its relationships with other relevant courses such as data warehousing and on-line analytic processing.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

© 2010 Higher Education Academy

Subject Centre for Information and Computer Sciences

40. AN ALGORITHM-BASED COURSE SYLLABUS

40.1 Course Overview

This section presents a syllabus for an introductory course on data mining. The syllabus is the result of constant improvement and evolution over the past 12 years. The purpose of the course is to teach data mining techniques and their effective applications to computing students. The course aims to achieve the following intended learning outcomes:

- To gain fundamental knowledge and understanding of basic data mining concepts and principles;
- To gain a broad understanding of different types of data mining problems and solutions;
- To gain in-depth knowledge and understanding of a selection of data mining solutions and appreciate their strengths and limitations;
- To gain understanding of quality and significance of data mining results;
- To obtain practical hand-on experience of applying data mining solutions to data sets.

Besides the intended learning outcomes, the course also provides an opportunity for students to gain practical key skills in using data mining software tools to perform data mining tasks. Other transferable skills from this course include analytic skills and problem solving skills.

The course assumes fundamental knowledge in data structures, algorithms, programming and databases. The prerequisite knowledge is normally obtained from courses at levels 4 and 5 in most UK universities. Basic understanding of time complexity of algorithms is needed to appreciate performance of data mining solutions. Familiarity with control structures and program functions is expected, although extensive programming experience in a specific language is not essential. Some elementary knowledge in probability and statistics is also assumed.

Due to the complexity of the subject, the prerequisite knowledge and academic maturity required, the subject is best taught as a level 6 course in the final year. The typical duration of the course is between 10 and 12 weeks.

40.2 Course Content

The main elements of course content are outlined as follows:

- Introduction (Backgrounds of information processing. What is data mining? Origins and current states of data mining. Data mining objectives. Promises and challenges);
- Principles of Data Mining (Data mining process and approaches. Data mining objectives and main categories of data mining problems. Information patterns and data mining solutions. Data and domain types. Forms of input data sets. Data sources. Data quality issues. Importance of pattern evaluation);
- Data Pre-processing and Exploration (Aggregation. Sampling methods. Dimension reduction. Feature selection and creation. Discretisation. Transformation. Handling of missing values. Use of summary statistics, visualisation and OLAP for data understanding);
- Basic Techniques for Cluster Detection (Problem overview. Measures of proximity. Basic clustering algorithms: K-means and agglomeration. Agglomeration schemes. Evaluation of cluster tendency and quality. Cluster interpretation);
- Other Techniques for Cluster Detection (Limitations of basic methods. Categories of cluster detection solutions. Density-based clustering (DBScan). Graph-based clustering (Chameleon). Model-based clustering (EM/GMM). Further issues with proximity measures. Clustering in practice);
- Decision Tree Induction for Classification (Decision tree and tree induction. Information gain and ID3 algorithm. Other attribute selection measures and algorithms (CART, CHAID). Model overfitting and tree pruning. Performance evaluation of decision trees. Decision tree in practice);
- Other Techniques of Classification (Nearest neighbour methods (PEBLs). Bayesian classifier methods (naïve Bayes). Rule-based methods (sequential cover). Artificial neural network. Comparisons of classification techniques. Ensemble methods for classification. Classification in practice);
- Mining Boolean Association Rules (Frequent itemsets and Boolean association rules. The Apriori approach for mining frequent itemsets and its limitations. The FP-Growth algorithm for mining frequent itemsets. Rule generation. Measures of strength for associations);
- Mining Techniques for Other Associations (Generalised and quantitative associations. Mining frequent itemsets for generalised and quantitative associations (Apriori). Issue of rule redundancy. Mining sequential patterns. Incremental mining of associations. Principles of parallel mining of associations);

- Data Mining in Practice (CRISP-DM Methodology. Data mining case studies. Business intelligence: putting data warehouse, OLAP and data mining into perspective. Survey of data mining tools. Data mining applications: web mining and text mining).

The coverage of the syllabus is meant to be wide, providing a broad picture on data mining with emphasis on three main streams, i.e. classification, clustering and association rule mining. The selected data mining solutions are limited to those that have matured over years of use and improvement. Course tutors can exercise certain degree of flexibility in selecting data mining solutions to be covered within the proposed framework in order to keep the course content relevant to the development of the technology. Theoretical content is only introduced within the context of data mining solutions. Coverage on statistical methods and artificial neural network are kept brief because each requires a separate course unit to cover properly.

40.3 Teaching, Learning and Assessment

The syllabus promotes a *white-box* approach in teaching the subject. The main focus of teaching is the working principles and performance of data mining algorithms, solutions, techniques and methods. Each key part starts with a description of the problems faced and input data, and then presents an algorithm or solution followed by illustration with a comprehensive example. After that, the performance of the algorithm is discussed. It should not be the case that equal amount of time and attention is given to every algorithm or solution without emphasis. The strategy is that basic solutions are fully explained and illustrated whereas only the principles of the advanced or alternative solutions are outlined.

The course content is normally covered in the exact order as presented. However, flexibility of a certain degree can also be exercised. The key parts on classification, clustering and association discovery can be taught in any order. Course tutors may also find it desirable to bring the CRISP-DM and case studies of the final part forward to the Principles of Data Mining part in order to accommodate a data mining project as coursework.

The main part of teaching and learning is classroom centred. Through lectures, the course tutor teaches the data mining techniques and the theoretical foundation underlying the techniques. Tutors mainly use tutorial sessions to show workings of the data mining techniques through examples and probe the boundaries of the techniques. It is therefore essential to arrange tutorial classes after the lectures.

To enhance student learning experience and to give the course a practical hand-on flavour, a data mining software tool should be adopted and used throughout the course. Therefore, practical classes for learning how to use the software must be arranged. The software tool gives tutors an opportunity to demonstrate the entire mining process of a type of information patterns examples. The tool also enables students to observe the mining procedure and practise data mining by using selected algorithms.

The understanding of the subject is assessed via coursework and written examination. The coursework should include a data mining mini project. Given a data set of certain size and complexity, students are asked to conduct a number of relevant data mining tasks on the given data set over a period of time. The project provides students an opportunity to go through every stage of the data mining process and make decisions and judgement over issues such as pre-processing, mining, evaluation and interpretation. The project aims to enhance the knowledge and understanding of the subject through a practical experience. The weight assigned to the project reflects the importance of the project.

41. COURSE DELIVERY, A PERSONAL EXPERIENCE

41.1 The Experience

The author first introduced a data mining course under the title “Information Discovery” (not quite appropriately) into Buckingham’s BSc Information Systems degree programme in 1997. The course was later recommended to the Computing Department of City University London. It was offered as a final year optional module there from 1998 to 2004. In 2007, the author helped the creation of a similar course known as “Data Mining and Data Warehousing” at Sarajevo School of Science and Technology (SSST) and lectured half of the course. That course has ever since become a final year core module in SSST. In 2000, a postgraduate version of the course was developed under the title “Applied Techniques of Data Mining” for Buckingham’s own MSc e-Commerce Technologies first and then for MSc Innovative Computing since 2006.

Students attending the courses in these institutions are from a wide range of backgrounds such as software engineering, computer science, computing, information systems and even business computing. The groups vary in size from small domestic groups around 10 at Buckingham to larger groups around 40 to 65 at City and SSST. Undoubtedly, many students have found data mining new and difficult to study. Despite their difficulties and differences in backgrounds, the overwhelming majority of students (normally more than 85%) have succeeded. Many of them have found the subject interesting and enjoyable. Since their graduation, some

students have even taken up jobs in data mining in the IT industry, and some have decided to further their interest in data mining by research for higher degrees.

In the early years, due to lack of suitable textbooks on the subject, courses had to be taught using research papers. This was hard for undergraduate students who had to rely almost solely on tutorial classes to gain better understanding of the subject. The situation has been greatly improved, at least to the course tutors, since the textbooks such as [3], [4] and [12] became available. It was until the publication of [7] that course tutors as well as students have had for the first time a comprehensive text covering a range of data mining issues. However, due to its wide coverage and theoretical content, the book may still be quite hard to use for many current undergraduate students. The lack of suitable texts for the mainstream undergraduate students is in strong contrast to the abundance of supply of texts on databases. It is the main reason that motivates this author to write his own textbook "Data Mining Techniques and Application, An Introduction" [2].

Finding and using a suitable software tool to assist the teaching was another problem encountered in early years before 2006. At the very beginning, this author used special-purpose software tools such as CBA and See5 because the alternative tools at the time were either too expensive to purchase or too cumbersome to use. In 2000, the author obtained a free licence for IBM Intelligent Miner for Data and used the software till 2005. Although being successful for a large number of real-life data mining projects, the software performance was not stable and efficient as expected under the author's own institutional client-server infrastructure. The tool was not very easy to learn and was not always well supported. The arrival of Weka Explorer [12] has finally changed the landscape. Since 2006, Weka has been used satisfactorily in Buckingham and later in SSST in support of the teaching and learning of the subject.

Setting the right coursework was also a challenge. In the earlier years when data mining tools were not widely available, it was difficult to set any practical coursework. The coursework at the time was mainly a collection of exercises that required students to practise the use of data mining techniques on artificial small data sets, gaining better understanding of the working principles and boundaries of the techniques. Implementing some basic data mining algorithms was also tried for students of computing majors. Since the use of Weka in 2006, a data mining mini project has become a compulsory and significant part of the coursework. A team of three students undergoes the entire lifecycle of data mining and presents their finding to the whole class. In order for students to do the project well, as another piece of earlier coursework, students are required to find a good case study from the ever-enriched data mining literature and observe how data miners of the real world conduct their data mining. The project idea has worked well so far. Predicting algae growth in rivers, profiling fruit and vegetable eating habits, and identifying abuse of email system are a few success stories.

41.2 Lessons Learnt

A number of lessons have been learnt from the experience. First, the course tutors should not always assume that the students in the class possess all the prerequisite knowledge required. Nowadays, it is not surprising for course tutors to find students in software engineering who have little idea about control structures and pseudo code, students in computing who have not come across time complexity of algorithms, and students in information systems who have never written any lines of program code. Nor should the course tutors be puzzled when students claim that they have never learnt logarithm. Probability and statistics that are essential for data mining, but it may well be true that computing students, particularly computer science students, have insufficient exposure to the knowledge since their school days. As a result, the course tutors usually find it unavoidable to provide additional lessons in probability and statistics within the course unit and within the course context. Tutors should try their best to ensure that the students follow their descriptions in expressions, formulae and algorithms.

Second, complete illustrative examples are extremely helpful and useful in showing the working principles of the data mining techniques. These examples, such as the calculation of information gain, normally use a small data set as input and *walk* through the entire algorithmic process. Such exercises are often time consuming and tedious, but they help students to better understand what really goes on behind a data mining decision. "No pain, no gain" is a motto that still bears some truth.

Third, a balance among data mining theory, principles and practice must be maintained. This is because the understanding of data mining techniques and the effective use of the techniques are closely related. Applying the techniques to realistic data sets from various domains of application will enable students to appreciate the strengths and limitations of the techniques and compare their performances. The balance is achieved through the use of a software tool. Weka provides a range of data mining techniques covered by the syllabus, making it a suitable tool to support the teaching.

Fourth, data mining is a huge and ever-growing field that was born out of a number of other related fields. No matter how competent a course tutor is, there bound to be data mining approaches, algorithms and methods that fall beyond the comfort zone of the tutor. This may partially explain why course tutors want to include more

teaching materials from their own fields of expertise, leading to hugely different versions of the course syllabus in different institutions. It is felt that the course tutor should try their best to give students a more balanced view of data mining. The course tutors must therefore be prepared to take the teaching of the course as an opportunity to learn and expand their areas of knowledge and expertise.

42. DISCUSSIONS

Teaching data mining to undergraduate students is a relatively new phenomenon. It is not surprising that there is little agreement in the community regarding what topics to teach and how to teach them. Despite repeated calls by the TLAD workshop organisers in recent years, contributions to the topic remain limited. There is an increasing degree of urgency now to have a comprehensive discussion over issues regarding the teaching, learning and assessment of the subject. This paper initiates the discussion by looking into a few fundamental points.

Black or White?

There have been debates regarding whether the subject of data mining should be taught by following a *black-box* or a *white-box* approach. The black-box approach treats a data mining solution as a black-box and only explains what the solution does rather than how it works. The white-box approach opens the box itself and explains not only “what a data mining algorithm does”, but also “how the algorithm works”. This approach fits well within the realm of computing related discipline areas where effectiveness and efficiency of algorithms are both of interest. Besides, the approach also has its practical merits in using the technology effectively. Nearly all data mining algorithms work with external parameters that are set by data analysts. The settings have a direct effect on the result patterns discovered. Without an insight on how the algorithms work, the data analysts will have little understanding on how to set the appropriate values for the parameters and will be unable to utilise the algorithms properly. The white-box approach has been proved effective by the author’s own experience. It is also the approach adopted by most existing textbooks except [1] and [5]. The black-box approach is only favoured if the aim of education is to achieve a general awareness of the technology rather than the development and practice of it.

A *hybrid* teaching approach based on both may also exist: the teaching of a data mining method can be done through two phases via separate lectures. In the first phase, the purpose of the data mining method and its parameters are thoroughly explained with regard to the input data and output patterns followed by an illustrative example to show the working of the method. In the second phase, the algorithm for the method is studied in detail and its performance analysed. Students from information systems and business computing backgrounds attend only the first phase whereas students in computing and software engineering attend both phases. This compromise may widen the appeal of the course to students of a range of backgrounds and make the course more viable.

Which Software Helps?

Using what software to assist the teaching can also be a debatable point. Existing tools on the market can be divided into commercial tools such as SAS Enterprise Miner, Oracle Data Mining, Microsoft SQL Server and SPSS Clementine, and free downloadable tools such as Weka and RapidMiner. The commercial systems are often cumbersome to learn and operate but are built to put up with the workload from real data sets of large sizes and high dimensionalities. If such a commercial system is already in use to support other courses such as database courses, learning is no longer a big issue, and adopting the system for data mining certainly has the *appeal* factor. In contrast, free downloadable tools are more lightweight, easy to learn and operate. For instance, Weka is such a free tool that offers a wide range of data mining solutions and data pre-processing facilities. Its Explorer module has an easy-to-operate GUI through which small-scale examples of data mining tasks can be demonstrated. The Knowledge Flow module can be used for batch processing of a more serious data mining task. Weka remains popular in the higher education sector and liked by many students.

In practice, it is often the case that one single data mining tool alone cannot always meet all requirements of the course. For instance, it has been realised that some data understanding and data pre-processing (e.g. discretisation) are better done in systems such as Microsoft Excel spreadsheet or SPSS rather than inside the mining tools such as Weka. The problem is normally caused by the weaknesses and limitations of the mining tools. Weka lacks of sophisticated visualisation tools and has its limitations in handling data sets of very large sizes. The software even has limitations in mining certain types of patterns such as Boolean association rules. The course tutors should be prepared to exploit transferable skills of the students from other courses to support the learning of this course.

Course Project: Merits vs. Difficulties

The author has argued strongly for the inclusion of a data mining mini project as an important part of the coursework. The project achieves the objectives of gaining practical data mining experience through the

process of data mining and enhancing the understanding of data mining techniques and their applications. The project is very similar to a database design project for a database course. It serves an important and irreplaceable role for the learning experience of the course. However, such a project also raises a number of concerns. First, the course tutors and students must be fully aware that the project is a *rehearsal* of data mining not a *real practice* of data mining. Therefore, the project experience should count much more than the usefulness of the final mining results. Misunderstanding the purpose of the project can lead to unrealistic expectation on students. Second, due to the uncertain nature of data and probabilistic nature of patterns, data mining project can be much harder to do than a database design project. Repetitive trials of data mining are expected before the best mining results are obtained. Third, for the very same reason, students need more guidance from tutors over the entire project period.

Positioning Data Mining

The course syllabus proposed in this paper is for a stand-alone course unit at level 6. However, in practice, the course is very likely to co-exist with a number of related courses on a degree programme. The related courses may include Applied Statistics, Machine Learning, Data Warehousing, Online Analytic Processing, Advanced Databases, etc. How to align the data mining course with the other related courses for the purpose of delivering a coherent body of knowledge is still very much an open question. In Buckingham, several attempts have been made. For the undergraduate, the course is aligned with advanced Database Technologies. For the postgraduate programme, data mining is aligned with Web Technologies and Image Processing where data mining techniques are applied to analyse web and image data. The positioning of data mining for the postgraduate seems more successful than that for the undergraduate.

43. CONCLUDING REMARKS

This short paper presented an algorithmic course syllabus for computing students that has matured over years of practice. Through the author's experience, the paper demonstrates that data mining, although complex, can be successfully taught to undergraduate students via the use of the white-box teaching approach. The paper suggests that practical course project that requires students to conduct a piece of data mining plays an important role in enhancing student's learning experience and assessing student's understanding of data mining techniques. The paper at the end urges for immediate discussions on major issues regarding the teaching of this subject within the database teaching community in order to reach some cross-university agreements over these issues.

44. REFERENCES

- [36] Berry, M. J. A. and Linoff, G., *Data Mining Techniques: For Marketing, Sales and Customer Relationship Management*, 2nd ed. Wiley Computer Publishing (2004)
- [37] Du, H., *Data Mining Techniques and Applications, An Introduction*, Cengage Learning: Andover (2010)
- [38] Han, J. and Kamber, M., *Data Mining: Concepts and Techniques*, 2nd Edition, Morgan Kaufmann Publishers (2006)
- [39] Hand, D., Mannila, H. and Smyth, P., *Principles of Data Mining*, The MIT Press (2001)
- [40] Larose, D. T. *Discovering Knowledge in Data, An Introduction to Data Mining*, John Wiley and Sons: New Jersey (2005),
- [41] London Metropolitan University, *MSc Data Mining*,
<http://www.londonmet.ac.uk/pgprospectus/courses/datamining-and-warehousing.cfm>, accessed on 21 May 2010
- [42] Tan, P.-N., Steinbach, M. and Kumar, V., *Introduction to Data Mining*, Addison-Wesley (2006)
- [43] The Quality Assurance Agency for Higher Education, *Subject Benchmark Statements, Computing* (2007)
- [44] The University of East Anglia, *MSc Knowledge Discovery and Data Mining*,
<http://www.uea.ac.uk/sci/study/science/postgraduates/cmp/MSCKDD-admiss>, accessed on 21 May 2010
- [45] The University of Greenwich, *MSc Data Warehousing and Data Mining*,
<http://www.gre.ac.uk/courses/pg/com/dwdm>, accessed on 21 May 2010
- [46] The University of Manchester, *COMP61011: Machine Learning and Data Mining*,
<http://www.cs.manchester.ac.uk/postgraduate/taught/programmes/acs/syllabus.php?code=COMP61011&year=2010>, accessed on 21 May 2010
- [47] Witten, I. H. and Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Ed. Morgan Kaufmann Publishers (2005)

INTRODUCING A FORENSIC FLAVOUR TO TEACHING DATABASES AT L5

Jackie Campbell
Senior Lecturer
Faculty of Information and Technology
Innovation North
Leeds Metropolitan University
J.Campbell@leedsmet.ac.uk

Sanela Lazarevski
Senior Lecturer
Faculty of Information and Technology
Innovation North
Leeds Metropolitan University
S.Lazarevski@leedsmet.ac.uk

ABSTRACT

The curriculum content of database modules generally engage students in a database software development cycle for relational databases and data warehouses, including writing SQL statements, application development and investigation into advanced database techniques such as data mining. These are often taught within case studies of purchase order systems, retail or health care systems. The current trend for crime scene investigation drama and the frequent stories in the news of personal tragedies involving incorrect data, missing data or data mix-up capture the attention of many. The truth is that crimes require data investigation to provide evidence and this requires databases, database development, analysis, knowledge and skill. As well as making case studies more attractive to students, we have in the computing area a new trend of computer forensics students, some of whom strongly believe that databases are not relevant to their study. Having case studies close to their field of work is an attempt to change these opinions.

This paper describes and evaluates the introduction of a 'forensic flavour' to the teaching of databases as part of an undergraduate Computing Degree to students in their second year of study. The 'forensic essence' involved introducing investigative and enquiry based learning techniques as well as selecting case studies based around real-life crimes and crime data. The aim and learning objectives, and curriculum content remained unchanged for the module. The initial findings are that the students engaged on average 30% better and enjoyed the experience more.

Keywords

Forensic case study, enquiry based learning, investigation, databases

45. INTRODUCTION

Databases are significant within any business and core to any computing related course (Date, 2008). Yet they are often perceived by students as an unnecessary part of the course and in our experience at Leeds Metropolitan University (LeedsMet) do not generate as much enthusiasm as the web-design based modules.

At LeedsMet there has been a rise in the number of students applying for forensic computing based courses. Many of the skills developed in the forensic courses such as investigation, problem solving and analysis are skills we would like to develop and encourage in all computing students. The media trend for 'Crime Investigation' television programs such as CSI (Crime Scene Investigation) may be having a positive effect on the number of the students applying to the Forensic courses. Databases are referred to and depicted in these programs and the data is crucial to the crime investigation as is the presentation of evidence. Databases figured significantly in the trial of Harold Shipman, a general practitioner in Hyde, Greater Manchester who was convicted in January 2000 of murdering 15 of his patients and of forging the will of one. For example:

- The statistics of deaths per surgery/area were analysed and compared to those where Harold Shipman practised.
- The quality of the data input by Harold Shipman into his systems was extensively analysed and found to contain many errors.
- The metadata on the database provided crucial evidence in the case. The records showed that the prescriptions had actually been created on a day inconsistent with the 'date prescribed' as on the prescription. (Professor Liam Donaldson, 2001)

The module feedback from the first cohort of computing forensics students to complete the DAD module indicated:

- The students did not perceive the relevance of databases to their course;
- Students had not engaged with the learning experience;
- They found working through the workbook for Oracle Apex monotonous.

To address these issues the team introduced a combination of enquiry based and investigative teaching techniques, and 'forensic' based case studies into the database curriculum at level 5 and level 6.

Our objective for the change are:

- To improve students' learning by promoting the importance of data and databases;
- To improve students' confidence, risk taking, investigation and problem solving skills by introducing self-led enquiry based learning;
- To improve the general perception of usefulness databases.

At level 5 we are confident that the later two have been achieved, while on level 6 all three.

This paper discusses how the 'forensic flavour' has been added to the L5 Database Application Development module (DAD) and the initial findings of the module evaluation.

46. THE DESIGN OF FORENSIC BASED ACTIVITIES

Database Application Development (DAD) is a L5 database module. It is core to all students on Computing and Information Systems awards. There are currently around 157 students studying the module. The DAD module is taken in Semester A, there is also an elective Databases Project Context module to run in Semester B at level 5.

The module builds on the database knowledge they have gained at L4, which introduces the basics of database design, SQL and application development using Oracle Apex. DAD taken in year 2 (level 5), aims to further develop their database skills from L4. The curriculum addresses advanced database models, logical and physical design stages, intermediate and advanced SQL and application development using Oracle Apex. If they choose to study Advanced Database Management A databases at level 6 (as an elective), they will study an advanced database concept - the data warehouse.

The content and learning outcomes of the module remain unchanged. To introduce the forensic aspect some of the activities have been replaced or modified to be based around crime case studies or use enquiry based learning techniques.

46.1 *Example: Enquiry based learning*

A partial case study was presented to students in the first tutorial session. They were asked to consider a database to record crime data regarding people, objects, locations and criminal events for the Police Forces of England and Wales. It is necessary to mention that when choosing the case study and setting the tasks, we considered the ethical appropriateness.

The task set was as follows:

Objective: Introduction of Enquiry based learning

Activity: Identify your requirements for the case study

The aim is to encourage an investigative approach and learning through enquiry.

Task:

1) Below are the essential requirements for the database system. You are also asked to include another 4 requirements that are relevant for the crime prototype database the Police are developing. These requirements have to be of some complexity. Please note that we are not interested in holding data concerning custody, intelligence, child abuse and domestic abuse — this is part of a different project.

The system should include the following:

- *The ability to record and retrieve crime location, time and identifying code.*
- *The ability to identify the police officer who recorded notes about the crime and also details of the person who reported the crime.*
- *The ability to identify the different types of crimes committed in each city. Crime can be either Primary Type or Secondary Type.*
- *The ability to retrieve crime data for each county and city.*
- *The ability to identify if any crimes are related; and if the offence includes more than one crime;*

- *The Police would like to be able to retrieve data on reported and non-reported crimes; (see Appendix 4)—this was an spread sheet screen shot with the actual police data (HomeOffice, 2010)*

Previously students were given a case study as a scenario. This is also common on modules at L4. By presenting students with only partial case study students were required to understand, investigate, specify requirements and 'own' a case study. To complete the above task they had to investigate how the Police store data, the type of data, how data is distributed and (in broad terms) what it normally includes. By 'owning' the case study students gained a better understanding of the given requirements and an ability to identify their own requirements. They found this a very challenging process: they were uncomfortable about receiving so few written instructions and did not have experience in gathering requirements. In the future, we plan to support this task with a workshop on identifying and gathering requirements.

47. DATA COLLECTION METHODS AND EVALUATION OF RESULTS

An empirical study was initiated using a paper-based survey for data collection methods, to critically evaluate the learning approach when using forensic and enquiry based activities. We aimed at meeting the first three levels of learning of Bloom's taxonomy (Bloom et al., 1956), that is 1) 'Recall of Knowledge', 2) 'Comprehension' and 3) 'Application' (i.e. the ability to apply learning to a new or novel task). We set out to achieve our three main objectives: to improve the general perception of databases; to improve student's learning by inspiring students with the importance of data (and databases) and to improve students 'Enterprise' (confidence, risk taking, investigation, problem solving) by introducing self-led enquiry based learning activities.

At the start and at the end of the module, a paper-based survey was distributed to all students to identify what they were expecting to learn and which skills they expected to develop on the module, such as Technical skills (SQL, DW, Apex), Investigative (data analysis, data integrity) and Enterprise skills (problem solving, personal time management, researching and scoping problem, communicating ideas, confidence and risk taking). These questions were directly linked with our set objectives for this study. Students could strongly agree, agree, neither, disagree or strongly disagree. The collected responses for the analysis purposes were grouped into agree, neither and disagree answers.

Additionally, the researchers hoped to gain an insight into the student perception prior to the delivery about their own learning, and then during and after the module delivery was completed. We also considered the possible impact of students filling questionnaires with similar questions pre and post delivery. Potentially this gave them a clear guidance as to what they were about to study. We believe that knowing what they will study will serve to support their confidence about the module. The time difference between both surveys was 14 weeks. We did not identify if the same people filled in the survey in the two occasions; therefore we cannot conclude on their individual progression.

The pre module survey included 51 out of 157 students on the module. The results show that only 41% expected to learn technical skills, 35% investigative skills, and 20% for enterprise skills. The rest of the results for all three sections were recorded as undecided. Our initial results indicated that we may have a group of students who were weak in learning and/or developing their database skills. Unfortunately these results were not analysed in detail immediately after conducting the survey. We had a goal and our objectives and did not take into consideration the motivation of the students. This paper considers level 5. However, for the purposes of analysis, we asked the same questions at level 6, where out of 44 students 19 filled in the questionnaire. The results show that 100% expected to learn technical and investigative skills, 79% for enterprise skills and 21% were undecided. This discrepancy is huge, these students had two full years of database knowledge and understanding, some of them had undertaken a database-based placement year as well.

The post module survey and feedback were conducted during the last session. Assessment demonstrations were also conducted during this session. It is possible that the module feedback given at this stage was affected by the grade they received at the time for the module. However, we believe our survey has not been affected because of the types of questions we asked. Another possible factor that could have affected the results is the number of questionnaires (module feedback surveys) they were asked to complete around that time. Again, because of the nature of the questions, we do not believe that this had an impact on our results.

The post module survey included 77 out of 157 students on the module. The results show that 90% felt they have gained technical skills, 80% investigative skills, and 75% enterprise skills. The rest was undecided. The results show a vast improvement. It is also worth mentioning that the results for the module engagement this year by 30% compared with last year and feedback was more positive than ever before.

In conclusion we believe that we have achieved what we set out to do. The results are satisfactory and the student's response shows that they want to learn and to be challenged. The success rate on the module was very high. For the next academic year we have a higher numbers of students choosing to do databases at

level 6 and we do believe that the personal satisfaction of their learning in this year and the team success in promoting databases and making course more relevant and interesting to Forensics students is partly responsible for this.

48. CONCLUSIONS AND RECOMMENDATIONS

Even though, it is difficult to measure learning (as discussed for example by Biggs, 1993), we can report that due to one or, more likely, a combination of the factors implemented in the 2009/10 delivery, the student experience was improved. Learning is a significant part of the student experience and the improved module feedback and student reflection, although not conclusive, does suggest an improved learning experience.

To conclude, the forensic flavour was appreciated by the students and it was felt by staff and students that there was a general interest in the application of crime based case studies to databases modules.

49. REFERENCES

- [1] Belbin, R. M., 2004, *Management Teams: Why They Succeed or Fail*, 2nd ed. Butterworth Heinemann, London.
- [2] Biggs, J. (1993) What do inventories of students' learning processes really measure? A theoretical review and clarification. *British Journal of Educational Psychology*, Vol. 63, No. 1, pp. 3-19.
- [3] Bloom, Benjamin, et al. (1956) *Taxonomy of Educational Objectives: The Classification of Educational Goals*. McKay, New York.
- [4] Bonk, C. and Graham, C.R., 2006, *The Handbook of Blended Learning: Global Perspectives, Local Designs*, John Wiley and Sons, San Francisco, USA.
- [5] Brown, M., 2005. chapter 12 Learning Spaces. In *Educating the Net Generation*. Retrieved 29 January 2009 from www.educause.edu/educatingthenetgen/ EDUCAUSE, Boulder, Colorado, USA.
- [6] Date C.J. (2004). *An Introduction to Relational Databases*. 8th ed. Addison Wesley.
- [7] Devlin, M., Drummand, S., Phillips, C. Marshall, L. (2008) Improving Assessment in Software Engineering Student Team Projects. Proceedings of the 9th Annual HE Academy - ICS conference, Liverpool Hope University, 26th Aug. - 28th Aug. 2008, pp. 127-133
- [8] Eckerson, W, 'Data Warehousing Special Report: Data Quality and the Bottom Line', Available at: <http://www.adtmag.com/article.aspx?id=6321&mp> [11 February 2010]
- [9] Gartner (2008), Available at: <http://www.gartner.com> [18th March 2008]
- [10] Hoffer, J.A; Prescott, M;McFadden, F, *Modern Database Management*, Prentice Hall 2007
- [11] HomeOffice 2010 Available at: <http://www.homeoffice.gov.uk/rds/ia/atlas.html> [11 February 2010]
- [12] Jenkins, T. (2005) How do they think they're doing? Proceedings of the 5th Annual HE Academy - ICS conference, University of York, 30th Aug. - 1st Sept. 2005, pp/ 39-43.
- [13] Kolb, D., 1983. *Experiential Learning*. Financial Times/Prentice Hall, Paramus, N.J., USA.
- [14] Low O'Sullivan, M. (2005) Authentic individual self-assessment, praxis and community. 28. Proceedings of the 2005 HERDSA Annual Conference. Sydney Australia 3 - 6 July.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

EXPLORING THE XML-RELATIONAL INTERFACE USING PROBLEM BASED LEARNING

Peter Lake

Department of Computing
Sheffield Hallam University
Furnival Building, City Campus
Sheffield, S1 1WB
p.lake@shu.ac.uk

ABSTRACT

In this paper, we describe the use of Oracle XML_DB technology in the context of an M-level module which explores the role of the Database Administrator (DBA) in general, and which focuses upon the various aspects of moving data between systems. The module adopts a Problem-based approach to Teaching and Learning and students are encouraged to explore the complexities around using XML as a medium for data transfer.

Keywords

XML, XML_DB, Oracle, Data migration, Data transfer, Databases

50. INTRODUCTION

Sheffield Hallam University have been running a Database Professional MSc which includes Oracle OCA certification training for the last five years. One of the requirements for the course is that we expose students to a variety of methods for passing of data between heterogeneous data sources.

As many of the students on the course wish to go on to become Oracle DBAs, we use the Oracle toolset to demonstrate this information exchange, whilst insisting that they review alternative tools as well. Traditionally we covered tools like Oracle IMP and Oracle EXP and especially Oracle SQLLDR. It has become apparent that XML is becoming the medium of choice for many data transfer jobs [2] and we needed to help our students explore this.

Physical Database Design is the module that most clearly identifies the use of XML as a means of data transport. It runs both in face-to-face mode, and in Distance Learning mode. In both cases we adopt a Problem Based Learning approach. Part of the problem presented was that data for updating core tables was available only in the form of XML from suppliers, and that certain management reports were needed in XML format - in other words, XML both in and out of Oracle.

As discussed in previous TLAD papers, practical sessions are an important stage in the teaching process since it enables students to apply their knowledge using enterprise-level database application development. We needed to create a tutorial which allowed students to use the Oracle XML handling tools regardless of their attendance mode. We ended up with a Blackboard-based tutorial which explored a variety of aspects of the relationship between XML and relational data.

Students, irrespective of the delivery mode, are exposed to the scenario at the heart of the problem based approach in the first week of the module. They are therefore encouraged to view the tutorial as a guide to how they may tackle the data transfer requirements set out in the assessment brief, but are further encouraged to explore alternative approaches and provide some evaluation and conclusions as part of the written element of the assessment.

51. DISCUSSION OF PROBLEM DOMAIN

Data movement between heterogeneous database systems is a vast subject in its own right. Practitioners suggest that the complexity of the process frequently arises due to incompatibility between the source and the target data structures [5].

Traditionally the intermediary medium would often have been the Comma Separated Variable (CSV) file, as this was well understood by database professionals. However, it has now become apparent that XML is becoming the *de facto* standard for the transfer of data, especially across the internet [2], because of its simplicity, Open Standard background and the ability to add semantics to the embedded data [9].

There are several business needs that can be addressed by the use of XML as an intermediary, including: moving data during system upgrade or porting from a legacy system; as part of a Data Warehousing solution;

or as part of a Business-to-Business application. In the module discussed below we consider both the former and the latter needs.

At the heart of the problem is the fact that XML data is hierarchical and non-normalised, whereas a database target is very likely to be relational and normalised. Moreover, in the case of inserting data into a relational table from an XML source, for example, there is a need to map the incoming data held within tags, to columns within specified table(s). This is made more difficult by the absence of any data-typing within the XML document.

52. ORACLE XML_DB

Since Oracle 8i, when there were just a few built-in PL/SQL functions and procedures which assisted with dealing with XML in the Oracle arena, Oracle have dramatically expanded the toolset available to DBAs who need to handle XML. In Oracle 10g, for example, they built a collection of XML tools into the core database, and branded that collection XML_DB [7]

Amongst the features that we discuss in the tutorial, and that can be used to meet the needs expressed in the assignment brief are:

- A new datatype called XMLTYPE which is an extension of the Oracle CLOB datatype. As well as storing the text, as a CLOB would, it allows the user to query the content using SQL or XQuery.
- Methods to allow the storage, query, and transformation of XML data while accessing it using SQL [6]
- Methods to perform XML operations on SQL data. [6]

53. MODULE RATIONALE AND OUTLINE

Much of our teaching in the computing area must, of necessity, have a technical focus. That said, particularly at M-level, we are expecting our students to be technically competent already. It is therefore in the areas of professional development and the broader reflective skills that we often concentrate our efforts. This paper will not, therefore, discuss the skills provision element, which we loosely describe as "training", other than to observe that for many professional training courses the delivery model is:

- Talk about it;
- Demonstrate it;
- Let them do it.

When using technologies in our M-level modules we would build on that basic model:

- Talk about it;
- *Foster discussion and debate*
- Demonstrate it
- *Ask for reflection*
- Let them do it
- *Encourage deeper reflection and evaluation*

This maps nicely back to Parker and Rubin [8], who argue that a curriculum should be organised according to the way knowledge is organised. They see four stages: Knowledge acquisition; Interpretation; Attaching Significance; Application.

However, our preference in terms of teaching styles is to adopt a Problem Based Learning (PBL) method, which itself extends the above approach. PBL is an effective way to contextualize learning in professionally focused education, leading the student towards meaning-making over fact-collecting [4]. This approach promotes "deep" rather than "surface" learning [3] and also prepares the students for the way that problems are likely to present themselves once they become practising professionals.

Assessment, too, is different with this approach. Of course it is important in the grading of the student, but the major purpose is to "...help the learner rehearse, remember and re-organise new material..[and to form] new memory links" [1].

Our approach to teaching in the Physical Database module is that we provide a real-world scenario of a legacy system which needs porting from Paradox into Oracle, and which should then be able to accept data from a variety of sources, including XML. The problem is explained in week one. There are some traditional tutorials and lectures towards the beginning of the semester, but then the students are left to try to use what they have learned to tackle the problems set out in the briefing document. In the latter stages we adopt a *teaching by*

wandering approach in which we regularly meet the students, discussing their progress, potential future developments, and assessing the manner in which they are addressing the requirements.

The marking scheme makes it clear that we are looking for the student to experiment; to try alternative ways of tackling the problems they face; to reflect upon and evaluate the results of those experiments; and to draw some conclusions about the strengths and weaknesses of the tools they try. Pertinent to this paper, they are asked to review a number of ways of getting XML data into the new database, using XML_DB tools, or any other method they choose.

Students are always asked to fill in an online questionnaire about the module. Sample sizes are very small, with only three students replying last year, but, when asked how well the Problem Based Learning approach worked for them, no student has ever answered on the negative side of the Likert scale.

54. USING XML_DB

In that the tools within XML_DB allow us to discuss all the major issues which surround the use of XML as a medium for data transfer, the decision to teach this technology seems a sensible one, especially given the focus on an Oracle career that many students have. That said, there were a number of issues that needed to be addressed before we could expect the students to engage fully.

5.1 PRIVILEGES ISSUE

The most significant issue when it comes to teaching database administration in our laboratories is that the students often require operating system privileges that are not granted to standard student accounts. With XML_DB, for example, there may be a need to create Directory Objects on the server. Not unsurprisingly the university's DBA will not allow any student write access to the server. On top of this, some of the functionality of XML_DB is built into DBMS_packages owned by SYS and expects output to go to the server terminal.

Several potential solutions were discussed with our technical support colleagues. To image the PCs in a lab in such a way that they allowed students administrator rights, and to have a local copy of Oracle installed was one idea, but laboratories are in short supply at SHU, so PCs could not be tied up in such a way, especially as such an image would only be required on one day of a week.

The selected solution was to use Virtual Machine (VM) Workstation software on each PC, and provide the students with a VM Image containing Oracle 11g on Windows XP for the duration of the module. This worked well. Students liked the *Snapshot* concept in VM that allowed them to save database contents and settings at regular intervals, thus providing restore points, meaning that they did not always have to go back to the empty database given to them at the start of the module when things went wrong.

5.2 XML FROM RELATIONAL

There are several ways to return well formed, valid XML from Oracle relational tables. In the two examples below we examine how to do this from an SQL statement and a PL/SQL stored procedure. Naturally the preferred solution will depend on the business requirement being met, and on how the data is stored (in pure relational tables, as Object Relational or as XML in a CLOB or XMLType).

5.2.1 SQL

Two extensions to standard SQL allow for generating XML from a SQL statement and they are explored in the tutorial:

- **XMLElement:** function transforms a relational value into an XML element. Eg:
`<element1>value</element1>`
- **XMLForest:** maps a relational set to a list, called a "forest", of XML elements.

5.2.2 Stored Procedure

Oracle introduced the XML/SQL Utility (XSU) in Version 8i, and the functionality has been improved in subsequent versions of Oracle. In the tutorial we use DBMS_XMLQuery. For example, these two lines generate a dataset (line 1) and then format into well formed XML (line 2):

Line 1: `Qry := DBMS_XMLQuery.newContext('SELECT * FROM CDS WHERE cddid=||to_char(cddid));`

Line 2: `output_xml := DBMS_XMLQuery.getXML(Qry);`

These lines would be part of a fuller stored procedure.

5.3 FROM XML TO RELATIONAL

Valid XML can become part of an Oracle database in a number of ways, and the method chosen will depend upon how the data is to be used once stored.

One scenario might be that we are merely storing the XML as it is, and allowing access to the data using the XQuery, functionality which comes when data is stored in a XMLType datatype. In this case, the XML file is read in as a CLOB and then re-cast as a XMLType. The key lines of code here are:

Line 1: `this_clob:=getclobfromfile('CMSPL4_ORACLE_WORK','NewCD1471.xml');`

Line 2: `INSERT INTO CDXML VALUES (xmltype (this_clob));`

A second scenario might be that we need to insert rows into an existing table from an XML file. One way of handling this would be to load this_clob (as line 1 above) and convert it to XMLType and then use the Extract method to search for values to create an INSERT statement with. For example, line 3 loads the variable Titl with whatever value is in the XML Tag TITLE from the XMLType variable called somexml:

Line 3: `Titl := somexml.extract('/ROWSET/ROW/TITLE/text()').getStringval();`

5.4 SUPPORTING LITERATURE

In comparison to other aspects of RDBMS technology, XML_DB is relatively new. This explains the relative paucity of literature available to support learning. There are the obvious sources on the Oracle website, including some detailed tutorials [7]. The reference we use with students is an Oracle Press book: Oracle Database 10g XML & SQL [9]. This is still not available in 11g version, so students are encouraged to review the "What is New in 11g" section of the XML_DB website.

55. CONCLUSION

Oracle XML_DB is a powerful and flexible tool for handling XML in a relational environment and, as such, it is a worthy example fit for teaching broader aspects of database functionality. The material available is still somewhat sparse, as this is relatively new technology, but with extra tutorial material to support their comprehension, and a context to work in, there is evidence that the important issues can be understood, debated and analysed by students.

56. REFERENCES

- [48] Cotton, J., *The Theory of assessment : an introduction*, Kogan Page. (1995)
- [49] Guha, S. et al, Integrating XML Data Sources Using Approximate Joins, *ACM Transactions on Database Systems*, 31(1), 161–207 (2006).
- [50] Iqbal, R., James, A.,, Scenario-based Assessment for Database Course, *6th Workshop on Teaching, Learning and Assessment in Databases*, Cardiff, UK, Higher Education Academy. (2008)
- [51] James, R., *Problem-Based Learning: An Introduction*, National Teaching and Learning Forum, 8(1), Available via http://www.ntlf.com/html/pi/9812/pbl_1.htm
- [52] MORRIS, J. *Practical data migration*. British Computer Society. (2006)
- [53] Oracle XML DB Developer's Guide (Web resource)
http://download.oracle.com/docs/cd/B28359_01/appdev.111/b28369/xdb01int.htm]
- [54] Oracle XML_DB home website: <http://www.oracle.com/technology/tech/xml/xmlldb/index.html>
- [55] PARKER, J. C. AND RUBIN, L. J., *Process as Content. Curriculum Design and the Application of Knowledge*. Rand McNally, Chicago, IL. (1966)
- [56] Scardina, M. & Chang, B. & Wang, J. *XML & SQL: Design, Build and Manage XML Applications*, Oracle Press. (2004)
- [58]

HANDLING XML DATA USING ORACLE 11g

Mary Garvey
University of Wolverhampton
School of Computing & IT
Wolverhampton
M.Garvey@wlv.ac.uk
<http://www.scit.wlv.ac.uk/~cm1958/>

ABSTRACT

This paper looks at the issues surrounding handling XML data using the features provided by a relational database management system (DBMS), Oracle 11g. To provide students experience of handling larger datasets, use of the DBLP Computer Science Bibliography dataset has been used. Loading the dataset has not been without problems, which as yet have not been fully resolved and will be discussed further in the paper.

Keywords

XML Data, relational data, Oracle 11g, DBLP Bibliography.

57. INTRODUCTION

Due to its non-propriety format and self-describing semantic structure, XML is becoming a standard information exchange language for business-to-business (B2B) and web-based applications. The ability to handle the amounts of data generated from such systems is crucial, in terms of efficiently storing and retrieving the information. Traditional data models, such as relational, are geared to handling fixed length, two-dimensional data, so XML data does not fit neatly into this model without modifications. Many companies however, will be using a relational database management system (DBMS) to handle their operational systems and may not necessarily want a separate native XML database (NXD) to handle the XML data, such as Apache Xindice (<http://xml.apache.org/xindice>) or eXist-db (<http://exist.sourceforge.net/>). Increasingly, relational databases, such as Oracle and IBM DB2 offer XML functionality to store and query the XML data, but can still be used as a relational database.

The use of XML has been introduced into some of the postgraduate database modules at the University of Wolverhampton. Currently the examples used in the tutorials focus on using the DEPT and EMP tables often used with Oracle databases. The downside to this is that students come away with the impression that a large database is something that is a bit more than 15 records. To give them experience of more realistic datasets, workshop exercises are being developed to store and handle XML data within a relational database, using the DBLP Computer Science Bibliography as an example of XML data [1], which contains 1.2 million publications [2]. The DBLP dataset also comes with a Document Type Definition (DTD) file that defines the structure of the data, which is also used in the workshop exercises.

The students are still introduced to XML concepts initially using the DEPT/EMP datasets, so not to overwhelm them, but the workshop will progress to handling the DBLP data to give them a better idea of handling larger datasets and the issues involved with transferring data from one system to another.

Oracle has been used as the vehicle for teaching databases at Wolverhampton for many years, so this paper concentrates on the different ways XML data can be stored and used in Oracle 11g. For example, the data can be kept in its XML format, using XMLType data types, or Character Large Objects (CLOBs), or converted into relational tables. Oracle also offers XML DB (<http://www.oracle.com/technology/tech/xml/xmlldb>), which provides native XML storage and retrieval facilities.

This paper will introduce the concepts covered in the workshop exercises, demonstrating the techniques used to store and query the XML data. For this academic year, the examples used in the postgraduate workshops were limited to the DEPT/EMP dataset and a subset of the DBLP data, with the examples based on the whole DBLP dataset prototyped with dissertation students only.

58. WHY USE XML?

XML is simple, flexible and platform-independent [3]: Simple, because it is a text-based format, which can be viewed and edited by a text editor such as Notepad, or vi. Though for large datasets generated from a DBMS, this would be unfeasible given the size of the files created; Flexible, because it allows users to define their own tags to mark up their data; Because of its non-proprietary format, XML is a cross-platform, software and hardware dependent tool, which can be used for transmitting and sharing information. Major vendors other than Oracle, such as Microsoft [4] and IBM [5] also offer support by adopting XML standards and making their products XML-enabled.

The advantages and disadvantages of XML are now well documented [6] and can be used in many applications areas, for example:

- **Data Exchange**
The Simple Object Access Protocol (SOAP) uses XML to encode messages, which is the standard for web service messages [7]
- **Application integration**
XML is used for integration within different applications; Microsoft, for instance, will use XML as the exchange format for its Office 12 products [8]
- **Content Management**
This involves handling different types of content on the web. XML, along with Extensible Style Sheets, can be used to handle and customize the layout of the content.
- **Data integration**
Combining data from different sources and presenting the user with an integrated view of the data.

59. XML SUPPORT IN ORACLE

Oracle provides different types of support for XML. For users with data in a relational format, XML data can be generated from within the database itself, which can be saved and exported as necessary, e.g., to a supplier in a business-to-business (B2B) setup. For example, to generate XML data from the DEPT/EMP schema, using SQL/XML Functions [9]:

```
SELECT XMLELEMENT(name "emp",  
XMLFOREST(e.empno, e.ename as empName, e.sal))  
FROM emp e;
```

Figure 3.1 Sample SQL/XML Functions

To import XML data there are several options, the XML document can be:

- imported into XML datatypes within the Oracle database;
- imported into Oracle's XML DB;
- stored in Character Large Objects;
- converted into relational format.

The first three approaches would be suitable for users who want to keep the data in XML format, though the first two should be used if using XQuery to manipulate the data.

The aim is to ultimately produce examples for the students to try all of these options. Three types of datasets will be provided and used in distinct stages:

- DEPT/EMP data familiar to Oracle users. This is used first, so they can absorb the concepts.
- Slightly larger dataset, comprising of the papers from researchers from the University (uniData.xml)
- Full DBLP dataset (dblp.xml).

60. LOADING THE DATASET

Each student can have their own copy of the DEPT/EMP data and the uniData.xml file, since these are relatively small.

Loading the entire dblp.xml dataset in one go proved infeasible. The University has a dedicated Unix Solaris 64-bit server for Oracle, but it was heavily in use by students at the time of import. The machine has 8GB of main memory, but quickly ran out of memory. The file therefore had to be split down into smaller chunks, in the end 12 smaller files were created, which proved not to be a strain on resources, plus had the added bonus that they could be viewed in the text editor vi. Each file had approximately 170,000 records. In total 2,131,046 records were added.

It is also infeasible for all students to have their own copy of the DBLP dataset, so a common “dblpUser” account was setup, which students are given read only access to. Dissertation students working on XML projects, however, can have their own copy if required.

To date, the dblp.xml has been imported in relational format and also loaded into the XML DB.

60.1 Loading the data in Relational Format

Java was used to load the data and each file took an average of 4 minutes each. Figure 4.1 is a snippet of the key code used to load the data and has been numbered for reference below.

```
1. public class LoadData {
2.     Connection conn = getConnection("dblpUsername","dblpPassword");
3.     OracleXMLSave sav = new OracleXMLSave(conn, "dblpTable");
4.     Reader xsltReader=new FileReader(new File('myStylesheet.xslt'));
5.     sav.setXSLT(xsltReader, null);
6.     sav.setIgnoreCase(true);
7.     sav.setCommitBatch(5000); }
```

Figure 4.1 Java code to load XML data

Key things to note:

Line 3: *dblpTable* is name of the table the data will be loaded into. The function does not create the table for you, this must be done previously.

Lines 4-5: attaches a style sheet with the data. This is vital if you want to store XML attribute data in the database too. For example, the following (figure 4.2) is a proceedings example:

```
<inproceedings mdate="2002-08-05" key="conf/iceis/GarveyJR02">
    <author>Mary Garvey</author>
    <author>Mike Jackson</author>
    <author>Martin Roberts</author>
    <title>Using Persistent Java to Construct a GIS.</title>
    <pages>257-262</pages>
    <year>2002</year>
    <booktitle>ICEIS</booktitle>
    <url>db/conf/iceis/iceis2002.html#GarveyJR02</url>
</inproceedings>
```

Figure 4.2 Example of DBLP Data

The *mdate* and *key* attribute data would normally be ignored if the style sheet was not attached. The type of reference could be important too, so the type is stored in a *name* attribute. In this case, it is an example of a conference reference (*inproceedings*). The following is a snippet of the style sheet that illustrates how the attribute and name data was retrieved:

```
<xsl:element name="name">
    <xsl:value-of select="local-name()" />
</xsl:element>
<xsl:element name="key">
    <xsl:value-of select="@key"/>
</xsl:element>
<xsl:element name="numbers">
    <xsl:value-of select="@number"/>
</xsl:element>
```

Figure 4.3 Style Sheet Example

Number is the name of an element in the DBLP data, which is a reserved word in Oracle and can not be used as an attribute name, so this also had to be renamed in the style sheet whilst loading the data.

Line 6: helped solve a problem with loading the associated dblp.dtd file.

Line 7: committed the records on a regular basis.

60.1.1 Relational Data Issues

Some elements are multi-valued, for example, authors in figure 4.2. When loading the data, only the last element is stored, so above, only Martin Roberts is accredited with the authorship. It is assumed this can be solved by further use of the style sheet.

The data is also in one 1NF table, further work would be needed to normalise this into a more appropriate schema.

60.2 Loading the data into XML DB

Loading the data into XML DB was quicker, with an average of 2 minutes each, though this was added at a later stage when the assessment work had finished and the author had the machine to herself. A future experiment will be made to compare both imports under similar circumstances.

This was loaded using PL/SQL, an example of the code to load this is:

```
BEGIN
    DBMS_XDB.CREATERESOURCE (
        abspath => '/public/demo/xml/uniData.xml',
        data     => BFILENAME('XMLDIR','uniData.xml') )
END;
```

Figure 4.4 Loading XML DB data

60.2.1 XML DB Issues

XML DB is accessed via a web page and browser support is not always consistent. If the data includes accents, such as áéù, then IE is needed. Firefox would report parsing errors, since it did not appear to pick up the DTD definitions. For example, IE would correctly show the author *José A. Blakeley*, but FireFox would say it is: *an undefined entity: José A. Blakeley*. Google's Chrome had similar problems too.

61. CONCLUSION AND FURTHER WORK

At Wolverhampton, the XML workshops have only been used on a postgraduate module, CP4009 Data Systems, which has an average of 110 students. The students have been practicing on the DEPT/EMP tables for a number of years, but this is the first year they have been introduced to the DBLP data. This year they have used the cut-down version of the dataset – the University researcher's data, which they can load into their own schemas. It is planned that in future they will also have read access to the whole DBLP data, currently this has been limited to dissertation students only.

It is vital though that they understand the principles first and introducing the dblp.xml dataset first would just overwhelm them, so the workshop needs to progress in stages. The researcher's subset has been well received; it gives a sample of most types of elements and is still small enough to read with an editor such as NotePad.

To conclude, the workshop is still very much in the preliminary stages and further work is needed to investigate loading the data into XMLTypes, plus examples of querying the data using XQuery. An evaluation of this teaching approach will then need to be taken, to see if students benefit from manipulating larger datasets.

62. REFERENCES

1. Ley, M., *DBLP XML records*. 2010, University of Trier.
2. Ley, M. *DBLP — Some Lessons Learned*. in *VLDB '09*. 2009. Lyon, France: Morgan Kaufman.
3. XML. *An Online Tutorial Guide to XML*. 2010 [cited 27 May 2010]; Available from: <http://www.xml-training-guide.com/>.
4. Microsoft. *XML - Data Developer Center*. 2010 [cited 27 May 2010]; Available from: <http://msdn.microsoft.com/en-us/data/bb190600.aspx>
5. IBM. *XML - Articles, online tutorials, and other technical resources on XML standards and technologies*. 2010 [cited 27 May 2010]; Available from: <http://www.ibm.com/developerworks/xml/>.
6. Sol, S. *XML Advantages & Disadvantages*. 2003 [cited 27th May 2010]; Available from: <http://www.theukwebdesigncompany.com/articles/xml-advantages-disadvantages.php>
7. W3C. *Simple Object Access Protocol (SOAP) 1.1*. 2000 08 May 2000 [cited 27 May 2010]; Available from: <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
8. Microsoft. *Ecma Office Open XML File Formats overview*. 2005 [cited 27 May 2010]; Available from: <http://www.microsoft.com/office/xml/licenseoverview.mspx>.
9. Robie, J. *SQL/XML Tutorial: SQL/XML, XQuery, and Native XML Programming Languages*. 2005 [cited 28 May 2010]; Available from: http://www.stylusstudio.com/sqlxml_tutorial.html.

TEACHING AND LEARNING DATABASES USING A GROUP CENTERED INTERACTIVE PROBLEM SOLVING APPROACH

Raman Adaikkalavan
Computer Science & Informatics
Indiana University South Bend
South Bend, IN, USA
raman@cs.iusb.edu

ABSTRACT

Teaching theoretical concepts and data modelling in an effective way that enhances student learning needs pedagogical changes. Additionally, balancing theory and practice is equally critical to the success of teaching and learning databases. In this paper, we discuss the group centered approach that we used in the undergraduate database course for Computer Science and Informatics seniors taught at Indiana University South Bend. This approach enhanced student learning at the same time had balanced theory and practice

Keywords

Group Centered Approach, Interactive Problem Solving, Undergraduate Database Course.

63. INTRODUCTION

Pervasive use of databases in various application areas is one of the major factors that attract large number of students to learn databases and related areas. Many students enroll in the database course with a notion that learning database means working with Structured Query Language (SQL). When introduced with logical data modeling, relation algebra, or normalization, some of the students feel overwhelmed. Not surprisingly, feedback from students toward the end of the semester, or after they have graduated and started working, is that those foundational theoretical concepts played a critical role in their success. It is clear that teaching databases with balanced theory and practice, and keeping students motivated at the same time is challenging. Thus, teaching databases to enhance student learning warrants innovative instructional methods.

In this paper, we discuss the group centered interactive problem solving approach we used in achieving the above mentioned goals in the undergraduate database course taught at Indiana University South Bend. This is the first course in database and consists of third year students from both Computer Science and Informatics departments and is offered as an upper-level elective. This course teaches database development whereas the second database course teaches database system internals. For the first course, we use “*Fundamentals of Database Systems*” [59] as the textbook. There are lots of research work [60][61] on teaching databases and discussion about the curriculum, however, none of them discuss the approach discussed in this paper. Some of the work in the literature are specific to a particular topic e.g., data modeling [62].

64. DATABASE CONCEPTS

In this section we discuss major topics that are covered in the database course and the approaches we used. We introduce and discuss each concept using real-world applications. After discussions, various types of assignments are given to students. Below we discuss each topic in detail.

HISTORY, CAREERS, DATABASE ARCHITECTURE: While discussing these topics we relate to a real-world system that students have used before. For example, *onestart.iu.edu* (student management system) is used to discuss about database architecture. This allows students to connect and be more interactive. We discuss about naive users, developers, administrators, customer relation management systems, enterprise resource planning systems, security and auditing, warehousing, mining, etc. This gives students a good overall view about the database and its related areas.

LOGICAL MODEL: We discuss both Entity-Relationship (ER) and Enhanced Entity-Relationship (EER) models, in detail. For each model we use examples from the textbook. After discussing each model an *individual in-class exercise* is carried out. For this exercise, a data model diagram is displayed on the screen. Each student

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

explains a specific part of the diagram. For example, a student explains a relationship between two entities. When someone makes an error or is not sure, other students get involved. This in-class exercise allows (even forces in a way) each student to take part in the discussion. The instructor takes the role of active listener. After discussing all concepts, *in-class group problem solving* session starts. The instructor provides two problems: 1) requirement specifications for creating a data model, and 2) set of data model diagrams with its requirements. Ad-hoc groups of two students are formed. Some groups work on the first problem (i.e., creating the logical data model for the given requirements.) Other groups work on the second problem (i.e., check whether the provided model captures the given requirements.) After the given time expires, one group (per problem) is invited to the instructor desk to explain their solution to the entire class using the blackboard. Other groups will be asked to help in-case of any issues. The same process is done for both ER and EER logical data models. These in-class problem solving sessions allow student to: 1) reinforce their understanding of concepts, 2) enhance learning by looking at how other students have solved the same problem, 3) learn from their mistakes, and 4) learn from other students mistakes.

RELATIONAL DATA MODEL AND MAPPING: We introduce relational model concepts and ask students to start doing the in-class exercise. For example, students identify keys, and update anomalies. We make sure that each student can contribute to a part of the problem. We discuss mapping of logical to relational model using the same ER and EER model examples. For the in-class exercise students are asked to do one of the mappings. After the discussion of relational data model and mapping, students are given a non-trivial logical data model diagram for the in-class group problem solving session. For example, the Ship tracking database diagram shown in Figure 7.8 on Page 237 of the textbook is assigned. This database involves weak entities, multiple identifying relationships for an entity, total participation constraint, and others. Ad-hoc groups of three students are formed and are asked to convert the logical to relational model. Finally, a volunteer group (if no one is willing, instructor picks one) is asked to explain the solution using the blackboard.

RELATIONAL ALGEBRA: For this topic we just do the in-class group problem solving session. After discussing all the relational algebra operators the problem solving session starts. The instructor selects a problem with multiple queries involving all the relational algebra operators. Each ad-hoc group is assigned multiple problems with one problem is solved by at least two groups. For example, exercise problem 6.16 from the textbook is used. The instructor also specifies additional conditions on how to solve the queries so that the problem solving session is streamlined. For the problem solving session, each group is asked to present their solutions if they have solved it completely differently. Usually, there are a lot of interesting discussions as the same query can be solved in different ways using different operators. More importantly, this allows students to learn the different approaches and allows them to discuss which approach is more efficient. Based on these discussions the instructor briefly discusses about query optimizations.

STRUCTURED QUERY LANGUAGE (SQL): The instructor provides a printout of a relational database schema along with a snapshot (with tuples). In all, 40 queries are discussed. For each query, the instructor discusses the requirement and asks students to find out the relations that need to be used, and how to solve the query. After the discussion, the instructor creates the SQL query and executes it in a relational DBMS. This allows students to understand how each query is executed. Sometimes students propose a different query to produce the desired results. Most of the queries are solved quicker except the division query via correlated sub-queries approach. Homework assignment follows the interactive lecture and is discussed later in the paper.

FUNCTIONAL DEPENDENCIES AND NORMALIZATION (1 TO BCNF): In-class group problem solving session following the lecture discussions involves closure computing, finding minimal FDs, equivalence between FDs, highest normal form that is satisfied by relations, and normalizing relations. Though not extensive as the other modules, during lectures, in-class exercises are done for each normal form separately. For example, students are asked to normalize a given relation so that it can satisfy the third normal form.

BASIC TRANSACTION PROCESSING AND OTHER ADVANCED TOPICS: In this module we discuss basic transaction processing by relating it to real-world applications such as airline passenger reservation. If time permits, advanced topics such as distributed databases are discussed.

65. DATABASE PRACTICE

In-class individual exercises and group problem solving sessions enhance student learning, however they do not provide them the necessary practical exposure. In order to accomplish this students do a term project and several homework assignments.

65.1 Term Project

The course has a term project that has been split into five major phases done by mostly a group of two students. For project selection we have tried two different approaches. In the first approach we asked students to select projects in their area of interests. List of suggested topics were also provided. In the second approach

students were given the choice of doing their own projects (e.g., for workplace), or select the project assigned by the instructor. In the second approach, more than 90% of the students selected the project assigned by the instructor. The major difference is that the first approach does not lend itself well for effective group centered classroom discussions. Though general discussions about each phase are possible, in-depth discussions about each project have to be done with the instructor outside the classroom. This approach reduces the interaction among students and prevents students to know how all the other students have solved a particular phase. For the second approach, instructor usually selects a real-world project. For example, iTest (discussed later) was selected as a project in 2009.

Group Centered Discussion and Learning: For each phase, before and after submitting, instructor and student groups have a lot of active discussion in the class. Some of the interesting submissions are displayed using a document camera or a ceiling projector, and are discussed in the class. The instructor provides a solution and it is refined in the class via multiple iterations. During the refinement phase students are split into ad-hoc groups of two or three. They are given a certain time. After that, instructor joins the discussion and each group is asked to present their refinements and justify them. This allows students to interact with student in their group and other groups and enhance their learning. These discussions via small learning communities are nearly impossible when the first approach is used for selecting projects.

Below we discuss each phase briefly and the tools used for the project.

- *Phase 1: Requirement Specification and Analysis* Students are asked to create a detailed description of the application - both from the requirements of what data needs to be represented/captured and what kind of computations (queries, reports, etc) will be made on the captured data. Students also are asked to write the description as detailed as possible so that the rest of the phases will be easier. In addition to the text and reference books, requirements for a sample project are posted as a reference.
- *Phase 2: Logical Data Model* Students capture refined requirements from phase 1 using the ER logical data model. Students are asked to identify entities (weak and strong), attributes (key, multi-valued, composite, derived, and NULL attributes), and relationships and constraints (total/partial, cardinality). For example, iTest project had more than 20 entities and M:N relationships with attributes. For capturing some of the requirements, EER was also used.
- *Phase 3: Logical to Relational Model Mapping* In this phase students convert the logical model from phase 2 to relational model. They also create SQL scripts to create relations, populate relations, delete tuples, drop relations, and “select *” queries for all relations.
- *Phase 4: Relational Algebra Queries* Students were asked to develop eight queries based on the application requirements discussed in phase 1. They were specifically asked to create two queries involving one relation, two queries involving two or more relations and four non-trivial queries involving three or more relations. Overall, queries must cover all of the following relational algebra operations: SELECT, PROJECT, RENAME, THETA JOIN, NATURAL JOIN, DIVISION, GROUPING and OUTER JOIN (Left, Right or Full), AGGREGATION, and UNION.
- *Phase 5: SQL Queries* All the relations created in phase 3 are checked for satisfying normal forms, dependency preservation, loss less join, etc in this phase. Students are asked to translate all the queries from phase 4 (after refinement) to SQL.
- *Tools Used:* We used the latest Oracle RDBMS for the project, and other freely available tools. ER data modeling tools such as Dia [63] and yEd [64]. Oracle SQL Developer and Data Modeler [65]. Dia and yEd are just data modelers, and cannot convert ER into relational model as opposed to Oracle data modeler. Though Oracle Data Modeler is useful it still does not support logical data model components such as UNION from the EER model. Oracle SQL developer and SQLPLUS were used to connect to the database. For populating data, we have used multiple approaches: sqlldr (SQL loader), Oracle SQL Developer, and SQL insert statements. Overall, GUI provided by Oracle SQL Developer was more useful and exciting to undergraduate students who were starting to learn database systems.

65.1.1 Sample Project and Beyond the Course

In this section, we discuss the iTest project developed as part of the database course, and its current status. The goal of iTest was to create a web-based testing system that allows students to take online tests. The uses would range from classroom tests, placement tests, surveys with detailed statistics in all the cases. It allows instructors to create question banks for each course or specialized topics and use those question banks while creating tests. It can be used as a tool for collecting and analyzing pre/post test data for courses such as computer literacy (used in general education), as well as capstone courses (used for program assessment). Since the data in iTest is stored in a database, longitudinal study and analysis becomes very simple. Changes

made to the program can be easily assessed and the results can be compared with prior results to see if recent changes to a course or the program have had a positive outcome.

Currently, an undergraduate student who completed the database course is developing the application. The system should be available for general use in late 2010. We have used open source technologies for the application development: PHP (scripting language for web programming), MySQL (database management system), JavaScript (a browser scripting language), CSS and HTML (to display the content), and SHA2 and AES (preserving data security).

65.2 Assignments

In addition to in-class group problem solving sessions and project, several assignments were included to enhance student learning. These assignments are handed out after solving problems in the class on each topic but before the project phase. Homework assignments include topics on designing architecture for real-world applications like *youtube.com*, data modeling, relational algebra, and normalization. With lab homework assignments, students were given a relation schema, data tuples, and ten queries (that use outer join, union, string matching, quantifiers, division, and aggregation) to manipulate the data. Students were asked to develop SQL scripts to create, delete, and drop relations, and a script with all the ten queries.

66. EXPLORATION

Data can be stored and manipulated using various data models. In order to impart knowledge about the current happenings in the research, industry, and elsewhere, we did two types of presentations.

Expert Presentation: We invited speakers from the local community to discuss the daily activities and issues they face as database developers or administrators. In 2009, we invited an experienced database administrator to come and give a talk. At the end of the session, students had lot of interesting questions and the feedback from the presentation was positive. One of the questions was: *is there a need to understand data modeling, relational algebra and normalization?*

Student Presentation: In order to allow students to explore the field of data and information management, we ask students to choose an emerging topic, and present it as a group of two. The presentation itself was done in several phases. In the first phase, every student group surveyed all the listed topics and selected three topics and ranked them. For each topic, they were required to write at least 4 major points that they plan to talk and send it to the instructor. After the instructor approved a topic, they were required to select three references that they want other students to read, and send it to the instructor. In the next phase, all the groups were required to read all the references, and write three questions they liked to ask the presenting group. This approach worked really well, and the presentation session was more active. In the last phase, students submitted their presentation, and gave their talk. Some of the topics that we used are as follows: Data Stream Management System, Main Memory Databases, Embedded Databases, Spatial Data Management, Massive Data Store with High Throughput, Database Security, and others.

67. STUDENT FEEDBACK

Overall, students rated the course as an excellent one that allowed them to learn database concepts and database programming. Some students rated this as the best course they have taken on their course evaluations and on their exit interviews. Few students felt that more time should be allocated to SQL.

68. CONCLUSIONS

In-class exercises and homework assignments allowed students to think individually. In-class problem solving sessions, presentations, and project phases allowed students to work as groups. Since the same type of task was repeated in multiple ways it allowed students to learn from their mistakes and others as well, and it also reinforced and enhanced their learning. In addition to these assignments and discussions, there were also unannounced quizzes and announced exams. In this paper, we discussed our approaches to teach databases with balanced theory and practice. Based on the student feedback it is clear that these approaches have enhanced student learning.

69. REFERENCES

- [59] R. Elmasri and S.B. Navathe, Fundamentals of Database Systems, 5th ed. Addison Wesley, Mar 2006.
- [60] S.W. Dietrich and S.D. Urban, "Database theory in practice: learning from cooperative group projects," in Proc of the 27th SIGCSE.1996, pp. 112–116.
- [61] E.S. Adams, M. Granger, D. Goelman, and C. Ricardo, "Managing the introductory database course: what goes in and what comes out?" in Proc of the 35th SIGCSE. 2004, pp. 497–498.

- [62] P. Wagner, "Teaching data modeling: process and patterns," in Proc of the 10th iTCSE. 2005, pp.168–172.
- [63] Dia - Diagram Creator, <http://live.gnome.org/Dia>.
- [64] yEd Graph Editor, <http://www.yworks.com>.
- [65] Oracle SQL Developer, <http://www.oracle.com/technology/products/database/sqldeveloper/index.html>

ERRORS IN ENTITY-RELATIONSHIP-DIAGRAMS: CLASSIFICATION AND STATISTICAL DATA

Martin Herzberg
Martin-Luther-Universität Halle-Wittenberg
Institut für Informatik
06099 Halle
martin.herzberg@informatik.uni-halle.de
<http://www.informatik.uni-halle.de/~herzberg/ER/>

ABSTRACT

Errors in Conceptual Database Design should be detected and removed early before the following design steps. Many common ER design tools (e.g. Oracle Designer) or general purpose design tools (e.g. GNOME Dia, Microsoft Visio) used for database teaching hardly offer any possibility of error recognition. We develop a tool (Database Designer) for Conceptual Database Design. The Database Designer will be used for teaching in basic database courses and database design courses. It can be used independently from any Database System in which the database may be implemented later. We study errors students make in database design and try to find possibilities to detect and avoid errors using this tool in our courses. In order to find feasible approaches and methods to achieve this goal, we analyzed database design proposals developed by students in exercises or exams. This paper proposes a classification of errors that can occur in ER diagrams and presents a list of all these errors together with statistical data of the occurrence of these errors in practice.

Keywords

Database Design, Entity-Relationship-Model, CASE-Tools, Error Classification, Error Statistics

70. INTRODUCTION

An Entity-Relationship-Diagram is a fundamental step at the beginning of every database or software engineering project. Of course, errors should be avoided as they affect each of following steps e.g. logical design, database implementation or implementation of application programs. We investigate errors in Entity-Relationship-Diagrams in order to help students building correct and stylistically good database models. This research may help to improve database teaching. As a first step, a categorization of these errors based on different criteria (e.g. cause, affected objects, effect) is proposed. On the basis of that we present a proposal for a list of the possible errors and their consequences. Errors in ER-diagrams can at first be categorized in syntactical, semantic and stylistic errors. Syntax errors violate the rules of diagram construction using the available constructs. The rules for the syntactical layout are well-known and therefore one can easily detect and avoid such errors. If one uses a suitable development environment or tool like Oracle Designer, CA ERWin or Sybase PowerDesigner, many errors that can occur in manual design, are precluded. An ER schema for a given application contains semantic errors if the schema is syntactically correct, but it does not represent the intended part of reality. The ER model might for example be too general (violates correctness) or too restricted (violates completeness). Stylistic errors are actually not real errors, but one can possibly express the relevant facts more easily. Complex models are much more error-prone. Meaningful naming of the constructs is very important for understanding ER models. So disadvantageous naming may constrict good understanding of the model and result in more errors. Based on this categorization and the studies of ER-diagrams created by students we present a list of possible errors in Sections 3 and 4. More detailed information can be found on the project's homepage at <http://www.informatik.uni-halle.de/~herzberg/ER/>.

71. RELATED WORK

Mentionable research already exists for some specific errors. Batra, Antony and Santhanam as well as Batra and Zanakis analyzed the cause of logical errors in database design in their work, e.g. [66], [67], [68]. They analyzed how students and designers approached database design and which influences could lead to errors. Dey et al. studied (amongst others in [69]) problems of relationships of higher degrees. Lenzerini and Calvanese as well as Hartmann investigated e.g. in [70], [71], respectively [72] inconsistent cardinality constraints and their application to other concepts of the ER-Model such as e.g. subtypes. Comprehensive research about classification of semantic errors in SQL can be found in [73] and [74].

72. SYNTAX ERRORS

Syntax errors can and should always be avoided. In some cases, it may be reasonable to allow certain syntax errors as an intermediate state during the database design process. Therefore we distinguish two groups of syntax errors. *Unacceptable Syntax Errors* can never be part of a reasonable intermediate state during database design and should always be avoided. Examples are missing links or objects placed at illegal positions. Formally spoken, this group contains all errors that cannot be corrected by adding other correct constructs to the model. *Temporary acceptable syntax errors* evolve during normal process while building up a diagram and could be accepted temporarily to make work easier. A *specialization without subentities* should for example be accepted temporarily to maintain a consistent top-down-design.

73. SEMANTIC ERRORS

Due to space restrictions we focus on groups of errors and leave out the details. For some specific errors we give the proportion of the affected student's works in brackets.

73.1 Structural Integrity

These errors are independent from the specific task. One does not need knowledge about the task to detect these errors.

73.1.1 Cycles and Inconsistent Dependencies

Relationship cycle with inconsistent cardinality constraints (3% of analyzed works affected) or similar errors like specialization cycles or weak-entity-relationship cycles belong to this group.

73.1.2 Entities

Entities are required to have unique name (1%), at least one attribute (15%) and a key must exist (17%).

73.1.3 Attributes of Entities

If a structured attribute has only one subelement the designer possibly misconceived this construct as concatenation of attributes. One has to introduce a new abstract superattribute and the former superattribute becomes simply another subattribute.

73.1.4 Keys

The most important errors in this group are non minimal keys (identifier inclusion), optional attributes declared as key and subsets of subattributes and the related abstract structured attributes declared as key. One can consider this key as non-minimal since there are further attributes that are part of the structured attribute but are declared as not contributing to the key.

73.1.5 Relationships

Common errors in this group are e.g. recursive mandatory relationship (11%) and recursive key relationships.

73.1.6 Exclusive Constraints (Arcs)

If an arc contains relationships with different minimum cardinality this equates to solely optional relationships. At most one of the relationships is instantiated. If an arc contains a relationship that is part of a key, every other relationship contained in this arc must also be an (alternative) key.

73.1.7 Specializations

Typical errors in this group are total specializations with only one subtype (9%) or implicit specializations (declared by type attribute and additional integrity constraints) (7%). Another interesting error are redundant specializations. If there is a direct specialization between two entities E_{super} and E_{sub} and in addition another arbitrarily long hierarchy of inheritance relations the direct inheritance is redundant. E_{sub} inherits all properties from E_{super} through the longer hierarchy.

73.1.8 Weak Entities

Naturally these errors also apply to association entities. Typical errors are wrong cardinality (not (1,1)) in key relationship (21%), missing partial keys in weak entities (1%) or bijective dependency (7%).

73.1.9 More Errors

Other groups of errors that were not presented as a group on its own include errors with inconsistent domain definitions, structured attributes or other constructs that belong to a specific notation only.

73.2 Semantic Integrity

This class of errors contains constructs which may be a correct solution for a specific task but an error for another task. Of course in theory one can find for each construct a task it does not apply to. Therefore we restrict the following list to the most important cases. These errors cannot be detected automatically since they are only errors in a certain context. Classification of these errors can be done analogous to *Structural Integrity* errors by affected objects, cause or effect of the error. We chose the latter approach. Errors are at first categorized by their effect. We consider database schemas S_{Error} and S_{correct} where S_{Error} is a given schema containing a semantic error and S_{correct} is a correct database schema with regard to the given task. Given a schema S , we use $I(S)$ to denote the set of possible states for the schema. Due to space restrictions we give only few examples for errors. All groups of errors and examples in detail can be found on the project's homepage.

73.2.1 Too Restricted Schema

The specified schema S_{Error} cannot represent all required states that are possible in reality modelled by S_{correct} i.e. $I(S_{\text{Error}}) \subset I(S_{\text{correct}})$. Typical examples are too restricted cardinalities of relationships (68%), mandatory instead of optional attributes (8%) or too general keys restricting the set of possible entities (23%).

73.2.2 Too General Schema

The specified schema S_{Error} can represent other states than are possible in reality modelled by S_{correct} i.e. $I(S_{\text{Error}}) \supset I(S_{\text{correct}})$. The most common errors were, analogous to *Too Restricted Schema*, too general relationship cardinalities (37%), too specific (restricting) keys (35%) or errors concerning exclusive constraints (13%).

73.2.3 Information not Representable

The most common examples were missing attributes (44%) and missing relationships (27%).

73.2.4 Redundant Information

Attributes were represented redundantly, e.g. foreign keys (34%) or inherited attributes. In some works students modelled redundant attributes instead of specialization (12%), specializations by indicator attribute (attribute-defined specialization) (8%) or simply unnecessary or already present constructs (22%, 45%).

73.2.5 Uncommon Cardinality

Examples are recursive mandatory 1:1 relationships (8%) or 1:1 relationships as implicit specializations (44%).

73.2.6 Errors on Weak Entities

E.g. key relationship declared as 1:1 mandatory (18%).

73.3 Style

The most frequent stylistic problems are naming problems (80%) and use of too complicated constructs (12%). Common examples are violation of naming conventions or too general or misleading names. Relationships are often given names like *has*, *belongs to* or *is*. Too complicated constructs are for example several equal relationships to all subtypes of a specialization instead of one relationship to the supertype.

74. ERRORS IN PRACTICE

74.1 Results

We studied the occurrence of these errors in practice. For this purpose exams and student's exercises were analyzed with regard to the occurring errors. The results were for the most part conforming to the results in the literature (see Section 71) and with our expectations. However, there were some unexpected findings on specific fields, such as for example modelling of derivable or redundant information. Entity types and attributes were correct in almost all works. Also keys and most relationships posed no problems as long as one did not have to consider relationships and keys for weak entities.

Structural Errors occurred very seldom. The most frequent error (21% of analyzed work affected) was an inconsistent cardinality specification for weak entities. In 11% of the samples students modelled structurally inconsistent recursive relationships. Other frequent structural errors were related to entity types.

The most frequent *semantic errors* are related to cardinality constraints. Too restricting constraints were found in 68%, too general constraints in 37% of the sample. Other frequent errors affected explicit and implicit specializations, too general and too restricted keys (23% and 35% resp.). We didn't expect to find that in many works some important constructs were simply missing.

In 44% attributes of entity types were not modelled and in 27% relationships were missing. A total of 101 works were analyzed. The table shown in Figure 1: Summary of analyzed work presents the summarised results of our studies. For each task we give the absolute number of errors in the first column and the number of affected works in the second column. All results in detail as well as the settings of the tasks and problems of the analyzed models can be found on the project's homepage.

74.2 Teaching

We found that many students made mistakes related to cardinality constraints and keys. As a consequence teaching in our basic database courses should possibly pay more attention to explaining these constructs. The *Database Designer* will help to improve database teaching. It supports Chen-Notation used in our basic database course and will support other notations (e.g. Barker's Notation, Martin Notation, IDEF1X) in the near future. As the *Database Designer* will be used in basic courses as well as in advanced database design courses we will be able to compare skills of our students and their learning progress easily.

Error Type	Number of Errors	Affected works (%)
Syntax	159	50
Structure		
Schema	4	4
Entity	45	22
Relationship	11	11
Specialization	21	13
Weak Entity	51	24
Semantic		
Too Restricted Schema	337	87
Too General Schema	211	62
Information not representable	195	63
Redundancy	243	67
Uncommon Cardinality	98	48
Weak Entities	30	18
Style		
Naming	338	70
Too Complicated Constructs	24	12

Figure 1: Summary of analyzed work

75. FUTURE WORK

To get a more comprehensive view on the errors that occur in praxis and teaching, more diagrams in different notations (others than crow's foot) and distinguished exercises should be analyzed. For this purpose, the *Database Designer* will be used for teaching in spring term 2010. Based on the results about the most common errors the *Database Designer* will soon provide feedback about e.g. questionable constructs like uncommon cardinality constraints or unusual specializations. The next step will be a prototypic implementation of error detection in the *Database Designer*. This implementation will serve as a basis for upcoming research on the assistance and detection methods proposed in [75].

76. REFERENCES

- [66] Antony S.R., Batra D., CODASYS: A Consulting Tool for Novice Database Designers, *ACM SIGMIS Database* **33**, 54-68 (2002)
- [67] Batra, D., Antony S.R., Novice errors in database design, *European Journal of Information Systems* **3**, 57-69, (1994)
- [68] Batra D., Zanakos S.H., A conceptual database design approach based on rules and heuristics, *European Journal of Information Systems* **3**, 228-239 (1994)
- [69] Dey D., Storey V.C., Barron T.M., Improving Database Design through the Analysis of Relationships, *ACM Transactions on Database Systems* **24**, 453-486 (1999)
- [70] Lenzerini M., Nobili P., On the Satisfiability of dependency constraints in entity-relationship schemata, *Proceedings of the 13th VLDB Conference*, 147-154 (1987)
- [71] Calvanese D., Lenzerini M., On the Interaction Between ISA and Cardinality Constraints, *Proceedings of the Tenth IEEE International Conference on Data Engineering*, 204-213 (1994)
- [72] Hartmann S., Coping with inconsistent constraint specifications, *Proceedings: Conceptual Modeling - ER 2001*, 241-255 (2001)
- [73] Brass S., Goldberg C., Semantic Errors in SQL Queries: A Quite Complete List, *Journal of Systems and Software* **79**, 630-644 (2006)

- [74] Goldberg C., Do You Know SQL? About Semantic Errors in SQL Queries, *7th International Workshop on Teaching, Learning and Assessment of Databases* (2009)
- [75] Herzberg M., Classification of Logical Errors in Entity-Relationship Diagrams (in German), *Diploma Thesis at Martin-Luther-University in Halle, Germany* (2007)

A COMPLETING APPLICATION FOR A DATABASE COURSEWORK ASSIGNMENT

Richard Cooper

Department of Computing Science

University of Glasgow

rich@dcs.gla.ac.uk

Monica Farrow

School of Mathematical and Computer Sciences

Heriot-Watt University

M.Farrow@hw.ac.uk

ABSTRACT

A database course typically requires for its practical work the student to develop a database together with a set of queries. Finishing off by having a database which can be accessed through the DBMS interfaces is all very well. Rather more motivating is to allow the student to finish off by adding a database application through which the database can be used. This paper describes a framework within the students can build a Java application which accesses their database. The use of this requires very little programming skills.

Keywords

database coursework design, database application.

77. INTRODUCTION

The ability to manage a database is a vital skill required of all computing professionals and so is an essential ingredient of all computing degree programmes. It is however not, for many strong students, the most engaging nor the most challenging aspect of the programme. Indeed, one of our colleagues feels the need to point out in the first lecture that, although this is the case, the database course is the one that will get a student a job.

There is some variety in course content, but the course mostly follows a standard pattern of database design, database creation, querying and a variety of supportive aspects of DBMS technology. Knowledge uptake will usually be tested by examination, but will also include coursework, which may be the piecemeal application of each technique taught in the course or, more usefully, the integrated use of those techniques by the development of database and its attendant queries.

It is important in the practical work to get the students to deal with fundamental technology using SQL for database creation and querying rather than by using a database GUI front end. This gives the students a considerable amount of work and if the end point is only being able to use SQL to manage the database, this can seem unsatisfactory. Only by making practical use of the database can student satisfaction be encouraged and, while this can be achieved by bringing application builders at the end, a more instructive approach is to get the students to build an application program on top of the database. This is a skill which will be important to them in their subsequent career.

This paper describes a technique of rounding off the student experience by allowing him or her to create a Java application to access the database with a GUI. The context of this is of students who are simultaneously learning Java, but with the understanding that the programming ability of much of the class will be minimal. Therefore the application is designed so that the student can produce it with modest programming skills. A framework is created as a set of Java classes, one of which includes as much of the metadata and queries as are needed for the application the student wants to create. The student's task is to amend this class suitably and then to run it against the database.

The paper describes the coursework and then the application structure and concludes with our experience of the student appreciation of the application.

78. SYLLABUS

The course is called Information Systems and Database and is a long one, compressing two previously courses into one. The Information Systems part introduces the students to spreadsheets, multimedia and basic XHTML programming and CSS, emphasising a clear structure, separating style from content and emphasising the layering of software, numerical management by computers and the use of generic software to cope with generic problems.

The database part of the course builds on this by showing how a DBMS provides the appropriate facilities for managing large amounts of long-term data. The syllabus includes ER modelling, relational database structure,

database creation in SQL or a GUI, relational algebra, SQL querying and updating, transactions, storage structures, concurrency, distribution, etc.

79. COURSEWORK STRUCTURE

The single assessed piece of coursework is the creation of a database for the application described below, the use of various parts of DBMS functionality and the creation of a report. The database is Oracle 10g and the front end is Aqua Data Studio.

The database to be built concerns University contacts with industrial collaborators. The specification follows as Figure 1:

We want to be able to keep track of the university's contact with industrial organizations - including, information about companies, their locations, activities, staff and so on together with details of which members of the department have dealings with them. For each company, we will identify one member of staff as our primary contact. Company staff will only be considered to be uniquely described within their company.

Typical relationships with the companies include: providing a course that company employees can attend (NB courses for students are not part of the exercise); jointly working on a project; consulting with the company; having a vacation job for a student; having a studentship from the company; staff having previously worked for a company; visits by individuals or teams from the company or from the university to the company, etc. Relationships may be with individuals in the company or with departments.

Typical queries might be:

- Who do we know at IBM, at which site are they and what are their interests?
- What projects are we involved in with which companies?
- Who in the department has worked for Apple?
- Which courses are currently on offer to which companies?

Typical users might be:

- Staff and students of the CS department
- Staff and students of the Business Studies department
- The publicity department of a local company
- The university finance office

There are a range of possible sets of information here. You should specify a significant subset of the possibilities and start your database design from there. Remember again that it is the range of concepts that you exercise that is important not the completeness of the database to be able to handle everything that could possibly be useful.

Figure 1 The Database Specification

The coursework involves achieving the following:

- i) Creating a conceptual schema in the form of an ER diagram.
- ii) Deriving a relational schema from the ER diagram.
- iii) Implementing this schema by using SQL commands.
- iv) Deciding on the storage structures you want for optimal use of your database.
- v) Populating the database with a set of typical data using a suitable mechanism.
- vi) Defining indexes which accelerate certain queries.

- vii) Defining specialised views which are appropriate to various sub-groups of users to focus their attention or to prevent them having unauthorised access to data.
- viii) Defining a range of SQL queries which could be used as the basis of "Canned Queries" for naïve users.
- ix) Defining some typical users and roles and assert the privileges for each for access to the tables and views.
- x) Creating an application as a Java program connecting to the database having a user interface defined in Swing.

The submission is a document starting with a report suitable for a non-technical client and continuing with the documentation of the outcomes of each of steps (i) – (x) above. The next section discusses step (x).

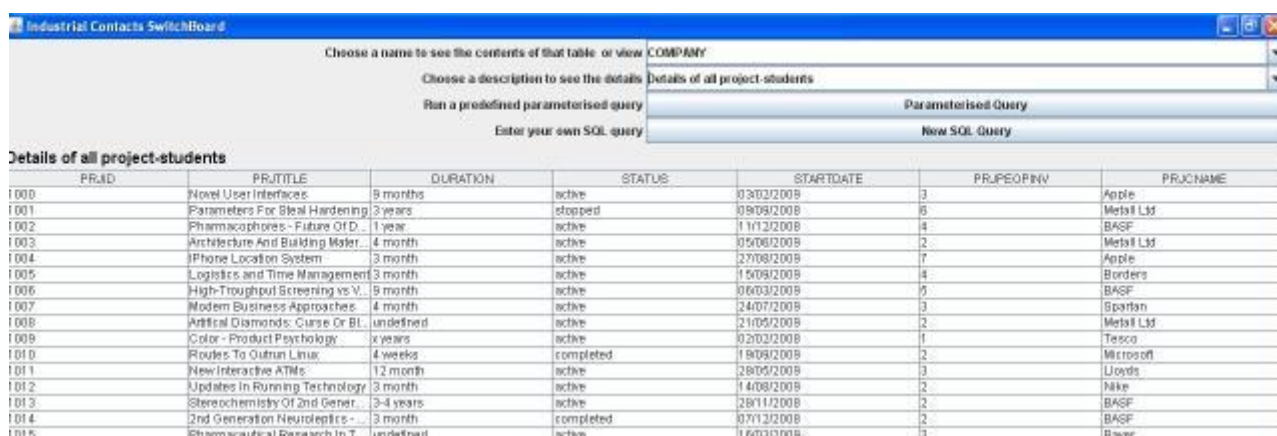


Figure 2 The Application User Interface

80. APPLICATION

80.1 The User Interface

The system provides an interface (shown in Figure 2) to the following functions:

- The user logs on with their Oracle username and password.
- A connection is made to the database.
- A main 'switchboard' is displayed with four methods of initiating queries at the top right of the screen. These give options to:
 1. view the data held in a specified table or view, choosing the name of the table or view from a combo box;
 2. view the data in a specified table or view, using a description instead of the name to make the choice, again from a drop down list.
 3. view the result of a simple query which has a single parameter;
 4. enter any SQL query and view the results.

The query output is displayed in the form of a table at the centre of the screen. If an SQL exception occurs, the details and the query causing the exception are displayed instead.

In the case of a parameterised query, another query is run automatically which produces a drop-down list of possible parameter values. When one of these parameters is chosen, the query is executed.

If the user chooses to enter an *ad hoc* SQL query, a text area is displayed in which the query can be entered. An 'execute' button is provided to run the query.

This is a Java application which can be run as normal using Eclipse. A dialog box is displayed for the user to log on, and a connection is made to the Oracle database. The main switchboard frame shows the options, and the data in a table or view is displayed in this frame. Separate frames are used for the parameterised queries and for a new SQL query. Each frame has a text area for feedback if an SQL Exception occurs. Note that there is an alternative version of the system which uses Servlets.

80.2 What the Student Has to Do

A collection of Java classes are provided which support the application. These do things like accept the login parameters, either for Access or Oracle, provide a JDBC interface to the database, display a JTable with the results, provide a pop-up to get query parameters, etc.

The only class which the student has to modify is one containing the metadata – *dbInfo*.

Figure 3 shows fragments of this class. The students have to modify the sections of the class which are embolden. These include the username and database name, a list of table and view names in the database, an array of pairings of descriptions and table names and an array of parameterised queries. Only the last of these needs much explanation. A parameterised query comprises: the parameterised query with “?” in place of parameter; a query description used for the menu and window title; a label for combo box of parameterised column values; and a query to get all possible parameter values.

```
class DBInfo
{
    public static String owner = "M09_rich";           //username
    public static String dbName = "Bank";              //database subject
    public static String tables [] =                    //table / view names
        { "ACCOUNT", "CUSTOMER", "OWNER", "EMPLOYEE", "BRANCH" };

    //Table/ View names and Description
    public static DescQuery descQueries [] =
        {
            new DescQuery ("CURRENTACCS", "Details of all current accounts"),
            new DescQuery ("FEMCUST", "List of female customers")
        };

    //PARAMETERISED QUERIES
    static String q0 = "Select AccountNo, Type, Balance, DateOpened "
        + "From " + owner + ".account where inBranch = ? "
        + "Order by Type, AccountNo";

    static String t0 = "Accounts at a particular branch";
    static String l0 = "Choose Branch number: ";
    static String p0 = "Select distinct branchNo from " + owner + ".Branch order by branchNo";
    ... etc.
    static OneParamQuery params [] =
        {
            new OneParamQuery(q0, t0, l0, p0),
            new OneParamQuery(q1, t1, l1, p1),
            new OneParamQuery(q2, t2, l2, p2),
            new OneParamQuery(q3, t3, l3, p3),
            new OneParamQuery(q4, t4, l4, p4)
        };
};
```

Figure 3 The *dbInfo* Class

81. EXPERIENCE

The application has been the final part of the course for around ten years. Feedback from the students suggest that they like it, even though those less able in programming are somewhat daunted by the prospect of doing this. Certainly, the structure then becomes a template for project work which involves a programmed application on top of a database. The major problem lies in creating the parameterised query but only because this involves the concatenation of several lines of SQL. It is all too easy for two lines to be run together without an intervening space, in which case the query is syntactically unsound.

Acknowledgements

We would like to acknowledge Raza Hassan and Calum Hutchinson for their initial versions of the software.

£40.00

ISBN 978-0-9565220-009

PUBLISHED BY:

**HIGHER EDUCATION ACADEMY SUBJECT CENTRE FOR ICS
FACULTY OF COMPUTING & ENGINEERING, UNIVERSITY OF ULSTER
NEWTOWNABBIEY, CO ANTRIM, N. IRELAND, BT37 0QB**

T: +44 (028) 9036 8020 F: +44 (028) 9036 8206

E: HEACADEMY-ICS@ULSTER.AC.UK

W: WWW.ICS.HEACADEMY.AC.UK